TIGHT LOWER BOUNDS FOR SHELLSORT

Mark Allen Weiss
Robert Sedgewick

CS-TR-137-88

February 1988

Tight Lower Bounds for Shellsort

Mark Allen Weiss

School of Computer Science
Florida International University
University Park
Miami, FL 33199

Robert Sedgewick

Department of Computer Science
Princeton University
Princeton, NJ 08540

ABSTRACT

Shellsort is a simple classic algorithm that runs competitively on both mid-sized and nearly sorted files. It uses an increment sequence, the choice of which can drastically affect the algorithm's running time. Due to the results of Pratt, the running time of Shellsort was long thought to be $\Theta(N^{3/2})$ for increment sequences that are "almost geometric", however, recent results have lowered the upper bound substantially, although the new bounds were not known to be tight.

In this paper, we show that an increment sequence given by Sedgewick is $\Theta(N^{4/3})$ by analyzing the time required to sort a particularly bad permutation. Extending this proof technique to various increment sequences seems to lead to lower bounds that in general always match the known upper bounds. This suggests that Shellsort runs in $\Omega(N^{1+\epsilon/\sqrt{\log N}})$ for increment sequences of practical interest, and that no increment sequence exists that would make Shellsort optimal.

March 15, 1988

Tight Lower Bounds for Shellsort

Mark Allen Weiss

School of Computer Science
Florida International University
University Park
Miami, FL 33199

Robert Sedgewick

Department of Computer Science
Princeton University
Princeton, NJ 08540

1. Introduction

Shellsort is a simple sorting algorithm proposed by D. Shell [15] in 1959. For nearly sorted or mid-sized files (a few thousand elements), Shellsort performs as well as or better than any known algorithm, including quicksort. Furthermore, it is an in-place sorting algorithm requiring little extra space and is easy to code.

Shellsort uses a sequence of integers $h_t, h_{t-1}, ..., h_1$ and works by performing insertion sort on subfiles consisting of elements h_i apart. We call this an h_i-sort. Thus, 1-sorting amounts to simple insertion sort while when a file is 2–*sorted*, all elements in even spaces and all elements in odd spaces are sorted, but there may be no relation implied between an arbitrary even-positioned element and an arbitrary odd-positioned element. Shellsort works by performing passes consisting of an h_t–*sort*, h_{t-1}–*sort*, and so on until an h_1=1–*sort*. It is both necessary and sufficient that some pass do a 1–*sort* for the algorithm to be guaranteed to sort a file. Typically the increment sequences used are "almost" geometric sequences with $h_k = O(\alpha^k)$ for some α, stopping with h_t being the largest integer in this sequence less than N. This is by no means a requirement, however these increment sequences perform better than others in practice.

Despite its simplicity, Shellsort has been analyzed only for some special cases. Furthermore, these results indicate that the dependence on the choice of increments can be dramatic. If t=1, then Shellsort is equivalent to insertion sort, an algorithm whose performance is well understood. For insertion sort, the running time is known to be proportional to the number of *inversions* in the input: the worst-case running

time is $O(N^2)$ [7]. For increments $1,2,...h_k=2^k,...$, originally proposed by Shell, Shellsort is again quadratic in the worst case, and $O(N^{3/2})$ on average. At the other end of the spectrum, Pratt gives a set of $O(\log^2 N)$ increments for which the running time is $\Theta(N\log^2 N)$, which is the best known bound for Shellsort. Unfortunately, it performs poorly in practice unless N is unrealistically large because there are too many increments. Between these extremes, new results have lowered the worst-case running time of Shellsort to values not quite optimal, but considerably better than quadratic. On the other hand not even the asymptotic growth of the average case performance is known, for the types of sequences used in practice, although none seem to be $O(N\log N)$.

In this paper, we consider lower bounds on the worst-case running time. The only previous nontrivial lower bounds for Shellsort are due to Pratt, who showed that for increment sequences of the form $1,...,h_k=c_1\alpha^k+c_2,...$, α an integer, Shellsort runs in $\Theta(N^{3/2})$ (subject to certain technical conditions). This property is held by most of the increment sequences that have been tried in practice, however Sedgewick [11] showed that if $h_k=4\cdot4^k+3\cdot2^k+1$, then the running time is $O(N^{4/3})$. Our first main result in this paper is to prove this bound is tight by constructing a permutation that takes the required time to sort. Incerpi and Sedgewick [5] have extended their results further reducing the upper bound to $O(N^{1+\epsilon/\sqrt{\log N}})$. Our second main result is to show that this bound is tight also, assuming that an unproven (rather fundamental) conjecture is true. Moreover, it appears that if the increments are of the form $h_k=\Theta(\alpha^k)$, then the bound of Incerpi and Sedgewick is the best possible.

Section 2 reviews the methods used to obtain the aforementioned upper bounds. This will eventually explain why the lower bounds and upper bounds match. In Section 3, we discuss the Frobenius pattern and prove a lemma about the number of inversions in this pattern. We use this lemma to prove the lower bound. In Section 4, we discuss generalizations of this result to other increment sequences. Open problems are discussed in Section 5.

2. Previous Upper Bounds

To derive upper bounds for Shellsort, we consider an old problem from number theory:

Suppose that a country wishes to issue only k different types of stamps. What is the largest postage

that can't be placed exactly on an (infinite-sized) envelope?

This is known as the Frobenius Problem, apparently because the mathematician Frobenius mentioned it often in his classes [1]. A more formal definition follows:

Definition: $g(a_1,a_2,...,a_k) \equiv$ the largest integer which cannot be represented as a linear combination, with non-negative integer coefficients, of $a_1,a_2,...,a_k$.

We make several simple observations: First, we assume $a_1<a_2<...<a_k$, without loss of generality. Throughout the rest of this paper, we shall make this assumption. Now, $g(1,...)=0$, as all positive integers are representable; we thus assume $a_1>1$. Also, we may assume that each a_i is independent of the other arguments (that is it cannot be represented as a linear combination of the other arguments) since otherwise it could be removed without affecting the result. Finally, $g(a_1,a_2,...a_k)$ is defined iff $gcd(a_1,a_2,...,a_k)=1$.

For $k=2$, we have a solution due to W.J. Curran Sharp [14]:

$$g(a_1,a_2)=(a_1-1)(a_2-1)-1$$

provided, of course, that a_1 and a_2 are relatively prime. The case for three arguments remained unsolved for quite some time, but in the interim Johnson [6] provided the useful result

$$g(a_1,a_2,...,a_k)=d\cdot g(\frac{a_1}{d},\frac{a_2}{d},...,\frac{a_{k-1}}{d},a_k)+(d-1)a_k$$

where

$$d=gcd(a_1,a_2,...,a_{k-1})$$

Several results in this period deal with special cases such as arithmetic or "almost arithmetic" sequences, but are seemingly useless for Shellsort analysis. Finally, Selmer [12] provided the formula:

$$g(a_1,a_2,a_3)\leq \max\left[(s-1)a_2+(q-1)a_3,(r-1)a_2+qa_3\right]-a_1$$

where

$$a_3\equiv sa_2 \bmod a_1, 1<s<a_1$$
$$a_1=qs+r$$

provided that the arguments are pairwise relatively prime and independent (if not, either Johnson's formula can be used, g is undefined, or one argument can be eliminated and Curran-Sharp's formula used).

Incerpi and Sedgewick [5] provided a lower bound for the Frobenius function:

$$g(a_1,a_2,...,a_k)\geq\Omega(a_1^{(1+1/(k-1))}),$$

provided of course that $a_1<a_2<...<a_k$.

We now need two important lemmas before we use the Frobenius function to upper-bound Shellsort. The first is the fundamental result for Shellsort:

Lemma 1: If a *k−sorted* file is *h−sorted*, if remains *k−sorted*.

Proof: See [4] or [7], or [10]. □

The implication of Lemma 1 is that when we come to h_k*−sort* a file, we know that it is already h_{k+1}*−sorted*, h_{k+2}*−sorted*, ... , h_t*−sorted*. Thus when we do insertion sort on a subfile and come to a particular element x_i, we know that there are many elements guaranteed to be smaller than x_i and so insertion sort shouldn't take as long as its worst-case bound would indicate. Lemma 2 tells us how many elements can be larger than x_i, and hence involved in the insertion sort:

Lemma 2: If a file is *h−sorted* and *k−sorted*, then for each i', $x_{i'-i}\leq x_{i'}$ whenever i can be expressed as a linear combination with non-negative integer coefficients of h and k.

Proof: If $i=sh+tk$, then $x_{i'}\geq x_{i'-h}\geq...\geq x_{i'-sh}$ since the file is *h−sorted*, and $x_{i'-sh}\geq x_{i'sh-k}\geq...\geq x_{i'-sh-tk}=x_{i'-i}$ since the file is *k−sorted*. □

Remark: Lemma 2 can be easily extended to take more passes into account.

From Lemma 2 (extended), it follows that if a file is $h_{k+1},h_{k+2},...,h_t$*−sorted*, then for any element x_i, the insertion sort needs to consider elements that are less than $g(h_{k+1},h_{k+2},...,h_t)$ away. Thus with Lemma 2, we have a way to bound the running time of Shellsort. For each h_k*−sort*, we bound the running time and then sum over all passes. The time to h_k*−sort* can be bounded in two ways:

(A) Since we are running h_k insertion sorts of size N/h_k, an obvious bound is $O(h_k(N/h_k)^2)=O(N^2/h_k)$.

(B) For each element x_i, only elements a distance $g(h_{k+1},h_{k+2},...,h_t)$ can be involved in exchanges and of these, only $g(h_{k+1},h_{k+2},...,h_t)/h_k$ can be involved in exchanges in an h_k–*sort*, since only every h_k*th* element is examined. Applying this bound to each of the N elements yields $O(N/h_k \cdot g(h_{k+1},h_{k+2},...,h_t)$.

As examples of this application,

Theorem 1: (Papernov-Stasevich) The running time of Shellsort is $O(N^{3/2})$ for the increments 1, 3, 7, 15, ... , 2^k-1 , ...

Proof: For $h_k > O(N^{1/2})$, we use bound (A) above; for smaller h_k, we use bound (B), taking into account only two previous passes. Since $gcd(h_{k+1},h_{k+2})\equiv 1$, $g(h_{k+1},h_{k+2})=O(h_k^2)$, so bound (B) is $O(Nh_k)$. Thus the running time, T, may be bounded by

$$T \le \sum_{h_k \le N^{1/2}} Nh_k + \sum_{N^{1/2} < h_k \le N} N^2/h_k$$

Since $2^t=O(N)$ (i.e. there are t increments less than N),

$$T \le N \sum_{1 \le j \le t/2} 2^j-1 \quad + \quad N^2 \sum_{t/2 < j \le t} \frac{1}{2^j-1}$$

which implies that $T=O(N2^{t/2})=O(N^{3/2})$. □

Remark: Due to Pratt's results, even if we take more passes into account in the Frobenius function, we can't lower the bound on T. It turns out that $g(h_{k+1},h_{k+2},...,h_t)=(h_{k+1})(h_{k+1}+1)-1=O(h_k^2)$.

Theorem 2: (Sedgewick) The running time for Shellsort is $O(N^{4/3})$ for the increments 1, 8, 23, 77, ... , $4^{k-1}+3\cdot 2^{k-2}+1$, ...

Proof: For all k it can be shown that h_{k+1},h_{k+2},h_{k+3} are pairwise relatively prime and for $k>2$, they are independent also. Using Selmer's formula with $s=4\cdot 2^{k-1}+7$, $q=2^{k-1}-1$, and $r=8$, one obtains

$$g(h_{k+1},h_{k+2},h_{k+3})=O(h_k^{3/2})$$

(The constant implied is 16). For increments smaller than $O(N^{2/3})$, we use bound (B), otherwise we use bound (A). Thus

$$T \le \sum_{h_k \le N^{2/3}} Nh_k^{1/2} + \sum_{N^{2/3} < h_k \le N} N^2/h_k$$

Again, if $2^t=O(N)$,

$$T \le N \sum_{1 \le j \le 2t/3} (4^{j-1}+3\cdot 2^{j-2}+1)^{1/2} \quad + \quad N^2 \sum_{2t/3 < j \le t} \frac{1}{4^{j-1}+3\cdot 2^{j-2}+1}$$

$$= O(N\cdot 2^{t/3}) = O(N^{4/3}). \square$$

Further improvements can be made by using different increment sequences that will allow more passes to be taken into account and lower the value of g. Typically, g is bounded by Johnson's formula. Using this technique, Incerpi and Sedgewick obtained bounds of $O(N^{1+\epsilon})$ and then $O(N^{1+\epsilon/\sqrt{\log N}})$ for Shellsort.

3. A New Lower Bound

In this section, we derive the main results of this paper by giving a permutation that is asymptotically as bad as possible for Shellsort. First, we define *inversions*:

Definition: Given a file of integers represented by $x_1, x_2, \ldots, x_N$, an inversion is any pair (x_i, x_j) such that $i<j$ and $x_i > x_j$.

As an example, the file 3,5,1,4,2 has 6 inversions, namely (3,1), (3,2), (5,1), (5,4), (5,2) and (4,2). In the worst case, the file is in reverse order and there are $\binom{N}{2}=O(N^2)$ inversions. Only a sorted file has no inversions, and on average a file has $N^2/4$ inversions. Exchanging two adjacent elements that are out of place removes exactly one inversion, so that insertion sort runs in time proportional to the number of inversions, and is thus quadratic in both average and worst-case.

To make our calculations easier, we will make the simplifying assumption that there are only two keys, 0 and 1. The following lemma then applies:

Lemma 3 (Swapping lemma): Swapping a 0 and a 1 a distance d apart in a 0-1 permutation removes exactly d inversions.

Proof: Consider an element between the 0 and 1 before the swap:

1...*e*...0

If e=0, one inversion is removed because e and the first element (1 before the swap, 0 after) will no longer be inverted after the swap. Similarly if e=1, then one inversion is removed because e and the last element will no longer be inverted. There are $d-1$ elements of this type. This plus the inversion that is removed by actually swapping the 0 and 1 accounts for a total of d inversions. This is all there can be since none of the elements to the left of the 0 or right of the 1 can be "uninverted". □

Remark: We can prove easily that if the elements of the permutation are not restricted to be 0 and 1, then the number of inversions removed lies between 1 and $2d-1$.

We prove later that using 0-1 permutations instead of general permutations does not affect our result. We will also prove our theorem only for a few specific permutation sizes, N, although again, our results are applicable for any value of N. We postpone details until later.

The natural permutation to consider is a file in reverse order. In our case this would be a file of $N/2$ ones followed by $N/2$ zeros (we assume that a file is sorted by lowest value on the left). This "natural" permutation is not a bad one for Shellsort, because the early passes quickly bring sortedness to the file. What we need is a permutation that is very unsorted to start with and not made nearly sorted by early passes.

The permutation we will use is closely related to the specific increment sequence and the Frobenius function. In particular, we have h_1=1 and $h_k=4^{k-1}+3\cdot 2^{k-2}+1$ for $k>1$. For any value of $k>1$, we choose $N_k=g(h_k,h_{k+1},...,h_\infty)+1$. Eventually, there will be some maximum $h_t<N_k$, and thus we may write $N_k=g(h_k,h_{k+1},...,h_t)+1$. If we store our permutation, P_k as $p_0,p_1,...,p_{N-1}$, then we define P_k as follows:

Definition: $p_i\equiv$ 1 iff i is representable as a linear combination in non-negative integer coefficients of $h_k,h_{k+1},...,h_t$ and 0 otherwise.

Remark: p_0=1 by the above definition, and this permutation can be thought of as the "Frobenius pattern" since it is so closely related to the Frobenius function.

Example: $N_2=g(8,23,77,\cdots)+1$=156 and

P_2= 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1

1 1 0 0 0 0 0 1 1 1 0 0 0 0 1 1 1 0 0 0 0 1 1 1 0 0 0 0 1 1 1 0 0 0 0 1 1 1 1 0 0 0 1 1 1 1 0 0 0 1 1 1 1

1001111110011111100111111101111111011111110

Our permutation has the following very desirable property:

Lemma 4: No exchanges are performed by Shellsort for the increments $h_t, h_{t-1}, ..., h_k$ on permutation P_k.

Proof: For any $h_{t'}$ such that $k \le t' \le t$, if $a_x=1$, then $a_{x+h_{t'}}$ must also equal 1, so that the lemma follows. □

This lemma shows us that the early passes do no sorting work at all for our permutation. We now show that P_k has a lot of inversions to start with, so that we can expect Shellsort to run slowly on it. We now need to estimate the number of inversions in our permutation. We start with the following lemma:

Lemma 5: $N_k = \Theta(h_k^{3/2})$.

Proof: We have

$$h_k=4^{k-1}+3\cdot 2^{k-2}+1$$
$$h_{k+1}=4\cdot 4^{k-1}+6\cdot 2^{k-2}+1$$
$$h_{k+2}=16\cdot 4^{k-1}+12\cdot 2^{k-2}+1$$
$$...$$
$$etc.$$

Divide the permutation into lines such that line l contains $p_{(l-1)\cdot h_k}$ to $p_{l\cdot h_k-1}$. In our example, we have

line 1: 1 0 0 0 0 0 0 0
line 2: 1 0 0 0 0 0 0 0
line 3: 1 0 0 0 0 0 0 1
line 4: 1 0 0 0 0 0 0 1
. . .

Consider a 1 on line $\alpha=2^{k-3}$. Its index clearly must have the form

$$\alpha\cdot 4^{k-1}+3\beta\cdot 2^{k-2}+\gamma.$$

Suppose that some element on line $\alpha=2^{k-3}$ with index $ph_k+qh_{k+1}+rh_{k+2}+sh_{k+3}+...$ is 1. Then the only possibility is

$$p+4q+16r+64s+256t+...=\alpha$$
$$p+2q+4r+8s+16t+...=\beta$$
$$p+q+r+s+t+...=\gamma$$

These three equations imply that $0 \le \gamma \le \beta \le \alpha$. Thus the number of 1s on line α is $|(\beta,\gamma)| \le \alpha^2$. It follows that

line $\alpha=2^{k-3}$ has at most 4^{k-3} ones. This implies that there are at least $15\cdot 4^{k-3}$ zeros on this line; thus, the Frobenius pattern does not end at this line. Since we have $\Omega(2^k)$ lines containing $\Omega(4^k)$ elements per line, the total number of elements is $\Omega(8^k)=\Omega(h_k{}^{3/2})$. Moreover, $g(h_k,h_{k+1},\ldots,h_\infty)\le g(h_k,h_{k+1},h_{k+2})=O(h_k{}^{3/2})$, proving the lemma. □

Lemma 6: The number of ones in the first half of the permutation, P_k is $\Theta(N)$.

Proof: Clearly the number of ones is $O(N)$, so we only need to show the lower bound. We partition the permutation into lines, as above and calculate the number of elements in the first 2^{k-4} lines that are expressible as a linear combination, in non-negative integer coefficients of h_k, h_{k+1}, and h_{k+2}. This is clearly a lower bound for the total number we need to show to establish the lemma. As in the proof of lemma 5, we have the three equations:

$$p+4q+16r=\alpha$$
$$p+2q+4r=\beta$$
$$p+q+r=\gamma$$

with $\alpha\le 2^{k-4}$ and we need to lower-bound $\left|(\alpha,\beta,\gamma)\right|$. Each triple (p,q,r) generates a unique triple (α,β,γ). To see this, note that if two triples (p_1,q_1,r_1) and (p_2,q_2,r_2) both generated the same triple (α,β,γ), then one of the three equations above would be redundant. It is easily verified that the three equations above are independent. Thus we only need to lower-bound the number of (*integral*) triples (p,q,r). The equation

$$p+4q=L$$

clearly has about $L/4$ solutions, so for each $0\le r\le\alpha/16$, there are about $(\alpha-16r)/4$ solutions. Thus, (with big-theta notation implied), for each line α.

$$\begin{aligned}\left|(\beta,\gamma)\right| &= \sum_{r=0}^{\alpha/16}\frac{\alpha-16r}{4}\\ &= \frac{\alpha^2}{64}-\sum_{r=0}^{\alpha/16}4r\\ &= \frac{\alpha^2}{64}-\frac{\alpha^2}{128}\\ &= \frac{\alpha^2}{128}.\end{aligned}$$

Thus each line $\alpha\le 2^{k-4}$ has about $\frac{\alpha^2}{128}$ ones. Thus there are

$$\sum_{\alpha=0}^{2^{k-4}} \frac{\alpha^2}{128} = \frac{8^{k-4}}{384} = \Omega(N)$$

ones in the first 2^{k-4} lines, proving the lower bound and hence the lemma. □

It is now easy to prove that this permutation has a quadratic number of inversions, which is the most we could hope for.

Lemma 7: The number of inversions in permutation, P_k, is $\Theta(N^2)$.

Proof: By lemma 6, there are $\Theta(N)$ ones in the first half of P, which implies an equal number of zeros in the second half. (See [8] for a quick proof of this). Thus there are $\Theta(N^2)$ inversions. □

Remark: The constant implied in this proof is quite small, because only the ends of P_k are considered. Empirical evidence suggests quite strongly that the number of inversions tends to $N^2/48$. Proving this would require a much tighter argument than the one above (which we have not done).

We are now ready to prove the first main result of this paper. The proof is actually quite simple, given all the lemmas that have already been shown.

Theorem 3: The running time for Shellsort is $\Theta(N^{4/3})$ for the increments $1,8,23,77,\ldots,h_k=4^{k-1}+3\cdot 2^{k-2}+1,\ldots$.

Proof: Theorem 2 establishes the upper bound, so we need prove only the lower bound. We run Shellsort on P_k, which has $\Theta(N^2)$ inversions to start with. Now no exchanges are performed during the h_t*–sort*, h_{t-1}*–sort*, ..., h_k*–sort*, and hence no inversions are removed during these passes. It follows from the swapping lemma that at most h_{k-1} inversions can be removed during any exchange. Thus the number of exchanges necessary is $\Omega(N^2/h_{k-1})$. We know that $h_{k-1}=\Theta(N^{2/3})$, hence we obtain the lower bound of $\Theta(N^{4/3})$, completing the proof. □

Remark 1: If we want to prove this for a general permutation of N integers, instead of for the case where the keys are restricted to be 0 or 1, we proceed as follows: Assign the largest integers to the 1s, and the smaller integers to the 0s. The particular order is unimportant. When we come to h_{k-1}*–sort*, we still have a quadratic number of inversions, and we can remove them only twice as fast as before. Hence the bound

still holds.

Remark 2: If we want to generate a bad permutation of arbitrary size N, where N isn't necessarily a Frobenius number (plus 1), take the next highest N' that is a Frobenius number of h_j , h_{j+1} , ... , h_∞ and use the middle N elements of P_j. This will have $\Theta(N^2)$ inversions. To see this, note that an extension of lemma 6 is certainly true because the number of ones on a line can only increase as the line number gets bigger. Thus there is a greater density of ones near the middle of the permutation than at the start. Lemma 6 implies lemma 7. Moreover, this permutation will also satisfy lemma 4, so we obtain an $\Omega(N^{4/3})$ lower bound for any N.

4. More Lower Bounds

The general technique used in the previous section can be extended to prove lower bounds for other increment sequences.

For instance, the increment sequence 1, 65, ..., $h_k=(2^k-3)(2^{k+1}-3)(2^{k+2}-3)= 8\cdot 8^k-42\cdot 4^k+63\cdot 2^k-27$, ... can be shown to make Shellsort run in $\Theta(N^{5/4})$ in exactly the same manner as above [13], [16]. The increment sequences of Incerpi and Sedgewick that yield $O(N^{1+\epsilon})$ upper bounds can likewise be proven tight.

In general, suppose the increments h_k satisfy $h_k=\Theta(\alpha^k)$ for any (not necessarily integer) α. Suppose that h_t is the largest increment and that we use the permutation P_k as before. In this case, we need to obtain the maximum value of h_k to use in generating P_k. $g(h_k,h_{k+1},\cdots,h_t)=N-1$ and thus the lower bound of Incerpi and Sedgewick implies that

$$h_k^{1+1/(t-k)}\le O(N).$$

On the other hand, we also have

$$h_k=\Theta(\frac{N}{\alpha^{t-k}}).$$

Combining these equations, we obtain

$$t-k\ge\Omega(\sqrt{\log_\alpha N})$$

which yields

$$h_k=O(N^{1-1/\sqrt{\log_\alpha N}}).$$

Thus, if P_k has $\Omega(N^2)$ inversions, we obtain a lower bound of $\Omega(N^{1+1/\sqrt{\log_\alpha N}})=\Omega(N^{1+\epsilon/\sqrt{\log N}})$ which

matches the best known upper bound for $O(\log N)$ increment sequences. We cannot prove that P has $\Omega(N^2)$ inversions, but we make the following conjecture which would be sufficient to prove this result:

Inversion Conjecture: Given $a_1<a_2< \ldots <a_k$, then the number of inversions in the Frobenius pattern (of size N) formed from these integers is $\Theta(N^2/k)$.

For $k=2$, the conjecture is easily proven. For other values of k, empirical evidence strongly suggests that the conjecture is true; in fact, the implied constant seems to be 1/24. Moreover, to obtain the lower bound for Shellsort, all we need is the following weak form of the inversion conjecture which must certainly be true:

Weak Inversion Conjecture: Given $a_1<a_2< \ldots <a_k$ then the number of inversions in the Frobenius pattern formed from the integers is $\Omega(N^2/f(k))$, with $f(k)=o(2^k)$.

We then have the following theorem:

Theorem 4: The running time for Shellsort is $\Omega(N^{1+\varepsilon/\sqrt{\log N}})$ for increments $h_k=\Theta(\alpha^k)$ for any $\alpha>1$ if the *weak inversion conjecture* is true.

Proof: By the discussion above, if the number of inversions is $\Omega(N^2/f(k))$, we obtain a running time of $\Omega(\frac{N^{1+\varepsilon/\sqrt{\log N}}}{f(\sqrt{\log_\alpha N})})$. If $f(k)=o(2^k)$, this is still $\Omega(N^{1+\varepsilon'/\sqrt{\log N}})$ for some $0<\varepsilon'<\varepsilon$. □

Remark: If the number of inversions is $\Theta(N^2/2^k)$, then we obtain the trivial lower bound of $\Omega(N)$ because $2^k=2^{\sqrt{\log_\alpha N}}=N^{1/\sqrt{\log_\alpha N}}$.

5. Conclusions and Open Problems

Increment sequences generally fall into two classes: *uniform and non-uniform.* Uniform increment sequences satisfy $h_k=f(k)$; the largest increment is determined by N. All of the increment sequences we have used are uniform. Non-uniform sequences are $h_k=f(N,k)$; an example is Gonnet's [2] $h_t=\lfloor \alpha N \rfloor$, $h_k=\lfloor \alpha h_{k+1} \rfloor$, $h_1=1$, with $\alpha=5/11$. Furthermore, to make Shellsort optimal, there can be at most $O(\log N)$ increments. There is no indication that non-uniform increment sequences will perform better than uniform

sequences either in theory or in practice and our results suggest quite strongly that no "almost geometric" uniform increment sequence will run in $O(N \log N)$. Since all seemingly reasonable $O(\log N)$ uniform increment sequences are "almost geometric", this seems to rule out $O(N \log N)$ Shellsorts for these types of sequences and thus, probably for $O(\log N)$ non-uniform sequences as well.

Some interesting open problems remain. First and foremost is proving our inversion conjecture, or any somewhat weaker form as suggested in Section 4. Assuming the inversion conjecture, proving that even if only some subset of increments is $\Theta(\alpha^k)$, then Shellsort is not $O(N \log N)$ would generalize our result quite a bit. This would take care of some uniform $O(\log N)$ increment sequences that don't strictly increase. It turns out that for many of these increment sequences, we can still prove the lower bound but we need a slightly different proof; a unifying concept would be nice. Finally, another interesting conjecture is that uniform and non-uniform increment sequences are asymptotically equivalent. Again, this would generalize our result quite a bit, and would probably lay to rest the question of whether or not Shellsort could be $O(N \log N)$ for some increment sequence.

6. References

1. A. Brauer, "On a Problem of Partitions", *American J Mathematics* **64** (1942), 299-312.
2. G. Gonnet, *Handbook of Algorithms and Data Structures*, Addison-Wesley, 1984.
3. T.N. Hibbard, "An empirical study of minimal storage sorting", *Communications of the ACM* **6** 5(1963), 206-213.
4. J. Incerpi, "A Study of the Worst-Case of Shellsort", Ph.D. Thesis, Brown University, 1985.
5. J. Incerpi and R. Sedgewick, "Improved Upper Bounds on Shellsort", *Proceedings 24th Annual Symposium on Foundations of Computer Science*, Tucson 1983, 48-55.
6. S.M. Johnson, "A linear diophantine problem", *Canadian J. Math.* **12** (1960), 390-398.
7. D.E. Knuth, *The Art of Computer Programming. Volume 3: Sorting and Searching*, Addison-Wesley, Reading, Mass. (1973).
8. A. Nijenhuis and H.S. Wilf, "Representations of integers by linear forms in nonnegative integers", *J. Number Theory* **4** (1972), 98-106.
9. A.A. Papernov and G.V. Stasevich, "A method of information sorting in computer memories", *Problems of Information Transmission* **1** 3(1965), 63-75.
10. V. Pratt, *Shellsort and Sorting Networks*, Garland Publishing, New York (1979). (Originally presented as the author's Ph.D. thesis, Stanford University, 1971.)
11. R. Sedgewick, "A New Upper Bound for Shellsort", *J. of Algorithms* **2** (1986), 159-173.
12. E.S. Selmer, "On the linear diophantine problem of Frobenius", *J. reine angew. Math.* **294** (1977), 1-17.
13. E.S. Selmer, "On Shellsort and the Frobenius problem", *unpublished manuscript*.
14. W.J. Curran Sharp, Solution to Problem 7382 (Mathematics), *Educational Times*, London (1884).
15. D.L. Shell, "A high-speed sorting procedure", *Communications of the ACM* **2** 7(1959), 30-32.
16. M.A. Weiss and R. Sedgewick, "On The Incerpi-Sedgewick Increment Sequences", *in preparation*, 1988.