A Lower Bound on the Average-Case Complexity of Shellsort

TAO JIANG

University of California, Riverside, California

MING LI

University of California, Santa Barbara, California

AND

PAUL VITÁNYI

CWI, Amsterdam, The Netherlands

Abstract. We demonstrate an $\Omega(pn^{1+1/p})$ lower bound on the average-case running time (uniform distribution) of p-pass Shellsort. This is the first nontrivial general lower bound for average-case Shellsort.

Categories and Subject Descriptors: E.4 [**Coding and Information Theory**]: *data compaction and compression*; E.5 [**Files**]: *sorting/searching*; F.1.3 [**Computation by Abstract Devices**]: Complexity Measures and Classes—*complexity hierarchies*; F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*sorting and searching*; I.1.2 [**Symbolic and Algebraic Manipulation**]: Algorithms—*analysis of algorithms*

General Terms: Algorithms, Performance, Theory

Additional Key Words and Phrases: Average-case complexity, computational complexity, Kolmogorov complexity, shellsort, sorting

---

1. *Introduction*

The question of a nontrivial general lower bound (or upper bound) on the average complexity of Shellsort (due to Shell [1959]) has been open for about four decades [Knuth 1998; Sedgewick 1996]. We present such a lower bound for p-pass Shellsort for every p.

Shellsort sorts a list of n elements in p passes using a sequence of increments $h_1, \ldots, h_p$. In the kth pass the main list is divided in h_k separate sublists of length $\lceil n/h_k \rceil$, where the ith sublist consists of the elements at positions j, where $j \bmod h_k = i - 1$, of the main list ($i = 1, \ldots, h_k$). Every sublist is sorted using a straightforward insertion sort. The efficiency of the method is governed by the number of passes p and the selected increment sequence $h_1, \ldots, h_p$ with $h_p = 1$ to ensure sortedness of the final list. The original $\log n$-pass[1] increment sequence $\lfloor n/2 \rfloor, \lfloor n/4 \rfloor, \ldots, 1$ of Shell [1959] uses worst-case $\Theta(n^2)$ time, but Papernov and Stasevitch [1965] showed that another related sequence uses $O(n^{3/2})$, and Pratt [1972] extended this to a class of all nearly geometric increment sequences and proved this bound was tight. The currently best asymptotic method was found by Pratt [1972]. It uses all $\log^2 n$ increments of the form $2^i 3^j < \lfloor n/2 \rfloor$ to obtain time $O(n \log^2 n)$ in the worst case. Moreover, since every pass takes at least n steps, the average complexity using Pratt's increment sequence is $\Theta(n \log^2 n)$. Incerpi and Sedgewick [1985] constructed a family of increment sequences for which Shellsort runs in $O(n^{1+\epsilon/\sqrt{\log n}})$ time using $(8/\epsilon^2)$ $\log n$ passes, for every $\epsilon > 0$. B. Chazelle (attribution in Sedgewick [1996]) obtained the same result by generalizing Pratt's method: instead of using 2 and 3 to construct the increment sequence use a and $(a + 1)$ for fixed a which yields a worst-case running time of $n \log^2 n (a^2/\ln^2 a)$ which is $O(n^{1+\epsilon/\sqrt{\log n}})$ for $\ln^2 a = O(\log n)$. Plaxton et al. [1992] proved an $\Omega(n^{1+\epsilon/\sqrt{p}})$ lower bound for p passes of Shellsort using any increment sequence, for some $\epsilon > 0$; taking $p = \Omega(\log n)$ shows that the Incerpi-Sedgewick/Chazelle bounds are optimal for small p and taking p slightly larger shows a $\Theta(n \log^2 n/(\log \log n)^2)$ lower bound on the worst case complexity of Shellsort. Since every pass takes at least n steps this shows an $\Omega(n \log^2 n/(\log \log n)^2)$ lower bound on the worst case of every Shellsort increment sequence. For the *average-case* running time, Knuth [1998] showed $\Theta(n^{5/3})$ for the best choice of increments in $p = 2$ passes; Yao [1980] analyzed the average case for $p = 3$ but did not obtain a simple analytic form; Yao's analysis was improved by Janson and Knuth [1997] who showed $O(n^{23/15})$ average-case running time for a particular choice of increments in $p = 3$ passes. Apart from this no nontrivial results[2] are known for the average case; see Knuth [1998] and Sedgewick [1996; 1997].

Results. Here we show the results given in the abstract, more precisely, Theorem 2. The main ingredient is Theorem 1. This is the first advance on the problem of determining general nontrivial bounds on the *average-case* running time of Shellsort.[3] The proof was originally obtained using Kolmogorov complex-

---

[1] "log" denotes the binary logarithm and "ln" denotes the natural logarithm.
[2] The trivial lower bound is pn comparisons since every element needs to be compared at least once in every pass.
[3] See, for example, Pratt [1972], Knuth [1998], Yao [1980], Incerpi and Sedgewick [1985], Plaxton et al. [1992], and Sedgewick [1996; 1997].

ity (for Kolmogorov complexity see Li and Vitányi [1997]). The idea is to consider an "individually random" permutation of the input list (a permutation incompressible in the sense of Kolmogorov complexity). If one encodes every move of Shellsort cheaply, and if the algorithm does not make a certain number of moves, then one obtains a too short encoding of the random permutation—contradicting the incompressibility of it.[4] Moreover, since the overwhelming majority of permutations is incompressible we obtain the bound on the average. It turns out that the argument can be translated to a counting argument. This we have done and present only the more elementary and shorter counting argument here. The original proof using Kolmogorov complexity is given in the preliminary version [Jiang et al. 1999]. It is instructive that thinking in terms of code length and Kolmogorov complexity enabled advances in this problem.

2. *The Lower Bound*

A Shellsort computation consists of a sequence of comparison and inversion (swapping) operations. We count just the total number of data movements (here inversions) executed. The lower bound obtained below holds a fortiori for the number of comparisons. The proof is based on the following intuitive idea: There are $n!$ different permutations. Given the sorting process (the insertion paths in the right order) one can recover the correct permutation from the sorted list. Hence, one requires $n!$ pairwise different sorting processes. This gives a lower bound on the minimum of the maximal length of a process.

THEOREM 1. *Let $0 < \epsilon < 1$ and n, p satisfy $p \leq (\epsilon \log n)/\log e$. For every p-pass Shellsort algorithm and every increment sequence, every subset of $n!/2^n$ input permutations of n keys contains an input permutation that uses $\Omega(pn^{1+(1-\epsilon)/p})$ inversions (and comparisons).*

PROOF. Let the list to be sorted consist of a permutation π of the elements $1, \ldots, n$. Consider a $(h_1, \ldots, h_p)$ Shellsort algorithm A where h_k is the increment in the kth pass and $h_p = 1$. For any $1 \leq i \leq n$ and $1 \leq k \leq p$, let $m_{i,k}$ be the number of elements in the h_k-*chain* containing element i that are to the left of i at the beginning of pass k and are larger than i. Observe that $\Sigma_{i=1}^{n} m_{i,k}$ is the number of inversions in the initial permutation of pass k, and that the insertion sort in pass k requires precisely $\Sigma_{i=1}^{n} (m_{i,k} + 1)$ comparisons. Let N denote the total number of inversions:

$$N := \sum_{k=1}^{p} \sum_{i=1}^{n} m_{i,k}. \tag{1}$$

CLAIM 1. *Given all the $m_{i,k}$'s in an appropriate fixed order, we can reconstruct the original permutation π.*

---

[4] Fix a Shellsort algorithm. Code the lengths of the inversion insertion paths in appropriate fixed order. Since the input permutation can be reconstructed from the coding, the overall length of the code must exceed the length of the shortest description of the input permutation. Since the latter is assumed to be incompressible this gives a lower bound on the sum total of the lengths of insertion paths and hence on the running time.

PROOF. The $m_{i,p}$'s trivially specify the initial permutation of pass p. In general, given the $m_{i,k}$'s and the final permutation of pass k, we can reconstruct the initial permutation of pass k. $\square$

Therefore, to every input permutation there must correspond a unique combination of N together with appropriate fixed order (say in lexicographical order of subscripts) of elements of a partition as in (1). How many such partitions are there? Choosing a elements out of an ordered list of $a + b$ elements divides the remainder into a sequence of $a + 1$ possibly empty sublists. Hence there are

$$D(N) := \binom{N + np - 1}{np - 1} \tag{2}$$

distinct partitions of N into np ordered nonnegative integral summands $m_{i,k}$'s.

Consider a subset S of $n!/2^n$ input permutations, and let the maximum number of inversions among them be M. Clearly, $M > 0$. Then, overestimating the number of partitions involved, $\Sigma_{N=0}^{M} D(N) \geq n!/2^n$ which implies $MD(M) \geq n!/2^n$. Then,

$$\log(MD(M)) \geq (\log n!) - n. \tag{3}$$

We know that $M \leq pn^2$ since every $m_{i,k} \leq n$. We have assumed $p < n$. Hence, $\log M < 3 \log n$. The standard estimate gives $\log n! = n \log n - O(n)$ for $n \to \infty$.

Estimate $\log D(M)$ by[5]

$$\log \binom{M + np - 1}{np - 1} = (np - 1)\log \frac{M + np - 1}{np - 1} + M \log \frac{M + np - 1}{M}$$

$$+ \frac{1}{2} \log \frac{M + np - 1}{(np - 1)M} + O(1).$$

The second term in the right-hand side is bounded as[6]

$$\log\left(1 + \frac{np - 1}{M}\right)^M < \log e^{np-1}$$

for all positive M and $np - 1 > 0$. Since $0 < p < n$ and $1 \leq M \leq pn^2$,

$$\frac{1}{2(np - 1)} \log \frac{M + np - 1}{(np - 1)M} \to 0$$

---

[5] Use the following formula [Li and Vitányi 1997, p. 10]

$$\log \binom{a}{b} = b \log \frac{a}{b} + (a - b)\log \frac{a}{a - b} + \frac{1}{2} \log \frac{a}{b(a - b)} + O(1).$$

[6] Use $e^a > (1 + \frac{a}{b})^b$ for all $a > 0$ and positive integer b.

for $n \to \infty$. Therefore, $\log D(M)$ is majorized asymptotically by

$$A = (np - 1)\left(\log\left(\frac{M}{np-1} + 1\right) + \log e\right)$$

for $n \to \infty$. Altogether, $A + \log M \geq n \log n - O(n)$. With $p \leq (\epsilon/\log e) \log n$ $(0 < \epsilon < 1)$ this can be rewritten as

$$(np - 1) \log\left(\frac{M}{np-1} + 1\right) \geq (1 - \epsilon)n \log n - O(n),$$

and further as

$$\log\left(\frac{M}{np-1} + 1\right) \geq \left(\frac{1-\epsilon}{p}\right) \log n - O\left(\frac{1}{p}\right).$$

The righthand side is positive and asymptotic to the first term for $n \to \infty$. Hence,

$$M = \Omega(pn^{1+(1-\epsilon)/p}). \qquad \square$$

THEOREM 2. *The average computation time (number of inversions, for $p = o(\log n)$, and comparisons, for $n/2 \geq p = \Omega(\log n)$) in p-pass Shellsort on lists of n keys is at least $\Omega(pn^{1+1/p})$ for every increment sequence. The average is taken with all lists of n items equally likely (uniform distribution).*

PROOF. Assume the terminology above. Take S to be the special set of $n!/2^n$ input permutations using the *least* number of inversions. Then, the number of inversions made by algorithm A for every permutation not in S is at least M in the previous proof. The theorem follows, since for $p = o(\log n)$ $(\epsilon(n) \to 0$ for $n \to 0$ in Theorem 1) the lower bound on the expected number of inversions of the sorting procedure is at least

$$\left(1 - \frac{1}{2^n}\right)\Omega(pn^{1+1/p}) + \frac{1}{2^n}\Omega(0) = \Omega(pn^{1+1/p});$$

and for $p = \Omega(\log n)$, the trivial lower bound on the number of comparisons is vacuously $pn = \Omega(pn^{1+1/p})$. $\square$

Our lower bound on the average-case can be compared with the Plaxton–Poonen–Suel $\Omega(n^{1+\epsilon/\sqrt{p}})$ worst-case lower bound [Plaxton et al. 1992]. Some special cases of the lower bound on the average-case complexity are:

(1) For $p = 1$, our lower bound is asymptotically tight: it is the average number of inversions for Insertion Sort.
(2) For $p = 2$, Shellsort requires $\Omega(n^{3/2})$ inversions (the tight bound is known to be $\Omega(n^{5/3})$ [Knuth 1998]);
(3) For $p = 3$, Shellsort requires $\Omega(n^{4/3})$ inversions (the best known upper bound is $O(n^{23/15})$ in Janson and Knuth [1997]);
(4) For $p = \log n/\log\log n$, Shellsort requires $\Omega(n \log^2 n/\log\log n)$ inversions;

(5) For $p = \log n$, Shellsort requires $\Omega(n \log n)$ comparisons. When we consider comparisons, this is, of course, the lower bound of average number of comparisons for every sorting algorithm.

(6) In general, for $n/2 > p = p(n) > \log n$, Shellsort requires $\Omega(n \cdot p(n))$ comparisons anyway since every pass trivially makes about n comparisons.

In Sedgewick [1997], it is mentioned that the existence of an increment sequence yielding an average $O(n \log n)$ Shellsort has been open for 30 years. The above lower bound on the average shows that the number p of passes of such an increment sequence (if it exists) is precisely $p = \Theta(\log n)$; all the other possibilities are ruled out.

3. *Conclusion*

The average-case performance of Shellsort has been one of the most fundamental and interesting open problems in the area of algorithm analysis. The simple average-case analysis of Insertion Sort (1-pass Shellsort), and similar analyses of Bubble sort, stack-sort and queue-sort are given in the preliminary version of this paper [Jiang et al. 1999] and serve as further examples to demonstrate the generality and simplicity of our technique in analyzing sorting algorithms in general. Some open questions are:

(1) Tighten the average-case lower bound for Shellsort. Our bound is not tight for $p = 2$ passes.

(2) Is there an increment sequence for $\log n$-pass Shellsort so that it runs in average-case $\Theta(n \log n)$?

REFERENCES

INCERPI, J., AND SEDGEWICK, R.   1985.   Improved upper bounds on Shellsort. *J. Comput. Syst. Sci. 31*, 210–224.

JANSON, S., AND KNUTH, D. E.   1997.   Shellsort with three increments. *Random Struct. Alg., 10*, 125–142.

JIANG, T., LI, M., AND VITÁNYI, P.   1999.   The average-case complexity of Shellsort. In *Proceedings of ICALP'99*. Lecture Notes in Computer Science, vol. 1644. Springer-Verlag, Berlin, pp. 453–462.

KNUTH, D. E. 1998.   *The Art of Computer Programming* Vol. 3: *Sorting and Searching*. 2nd ed. Addison-Wesley, Reading, Mass.

KOLMOGOROV, A. N.   1965.   Three approaches to the quantitative definition of information. *Probl. Inform. Trans. 1*, 1–7.

LI, M., AND VITÁNYI, P. M. B.   1997.   *An Introduction to Kolmogorov Complexity and Its Applications*, 2nd ed. Springer-Verlag, New York.

PAPERNOV, A., AND STASEVICH, G.   1965.   A method for information sorting in computer memories, *Problems Inform. Transmission*, 1:3, 63–75.

PLAXTON, C. G., POONEN, B., AND SUEL, T.   1992.   Improved lower bounds for Shellsort. In *Proceedings of the 33rd IEEE Symposium on Foundations of Computer Science*. IEEE Computer Science Press, Los Alamitos, Calif., pp. 226–235.

PRATT, V. R.   1972.   *Shellsort and Sorting Networks*. Ph.D. dissertation. Stanford University, Stanford, Calif.

SEDGEWICK, R.   1996.   Analysis of Shellsort and related algorithms. In *Proceedings of the 4th Annual European Symposium on Algorithms*. Lecture Notes in Computer Science, vol. 1136. Springer-Verlag, Berlin, Germany, pp. 1–11.

SEDGEWICK, R. 1997. Open problems in the analysis of sorting and searching algorithms. In *Proceedings of the Workshop on Probability Analysis of Algorithms* (Princeton, N.J.). http://www.cs.princeton/~rs.

SHELL, D. L. 1959. A high-speed sorting procedure. *Commun. ACM 2*, 7 (July), 30–32.

YAO, A. C. C. 1980. An analysis of $(h, k, 1)$-Shellsort, *J. Algorithms 1*, 14–50.