A Lower Bound on the Average-Case Complexity of Shellsort

Tao Jiang [*] Ming Li [†] Paul Vitányi [‡]

September 21, 1999

Abstract

We give a general lower bound on the average-case complexity of Shellsort: the average number of data-movements (and comparisons) made by a p-pass Shellsort for any incremental sequence is $\Omega(pn^{1+1/p})$ for every p. The proof is an example of the use of Kolmogorov complexity (the incompressibility method) in the analysis of algorithms.

1 Introduction

The question of a nontrivial general lower bound (or upper bound) on the average complexity of Shellsort (due to D.L. Shell [14]) has been open for about four decades [5, 13]. 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 [14] uses worst case $\Theta(n^2)$ time, but Papernov and Stasevitch [9] showed that another related sequence uses $O(n^{3/2})$ and Pratt [11] 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 [11]. 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 [2] 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 [12]) 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, Poonen and Suel [10] 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 [5] showed $\Theta(n^{5/3})$ for the best choice of increments in $p = 2$ passes; Yao [15] analyzed the average case for $p = 3$ but did not obtain a simple analytic form; Yao's analysis was improved by Janson and Knuth [3] 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 are known for the average case; see [5, 12, 13].

[*] Department of Computer Science, University of California, Riverside, CA 92521. Email: jiang@cs.ucr.edu. On leave from McMaster University. Supported in part by NSERC and CITO grants, and a UCR startup grant.

[†] Dept of Computer Science, Waterloo, Ontario N2L 3G1, Canada. Email: mli@math.uwaterloo.ca. Supported in part by NSERC and CITO grants and Steacie Fellowship.

[‡] CWI, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands. Email: paulv@cwi.nl. Supported in part via NeuroCOLT II ESPRIT Working Group.

[1] "log" denotes the binary logarithm and "ln" denotes the natural logarithm.

Results: We show a general $\Omega(pn^{1+1/p})$ lower bound on the average-case running time of p-pass Shellsort under uniform distribution of input permutations for every p. [2] This is the first advance on the problem of determining general nontrivial bounds on the *average-case* running time of Shellsort [11, 5, 15, 2, 10, 12, 13]. Another goal of this paper is to demonstrate the use of Kolmogorov complexity (the incompressibility method) in the analysis of algorithms.

Kolmogorov complexity and the Incompressibility Method: The technical tool to obtain our results is the incompressibility method. This method is especially suited for the average case analysis of algorithms and machine models, whereas average-case analysis is usually more difficult than worst-case analysis using more traditional methods. A survey of the use of the incompressibility method in combinatorics, computational complexity, and the analysis of algorithms is [6] Chapter 6, and recent work is [1, 8].

Informally, the Kolmogorov complexity $C(x)$ of a binary string x is the length of the shortest binary program (for a fixed reference universal machine) that prints x as its only output and then halts [4]. A string x is *incompressible* if $C(x)$ is at least $|x|$, the approximate length of a program that simply includes all of x literally. Similarly, the *conditional* Kolmogorov complexity of x with respect to y, denoted by $C(x|y)$, is the length of the shortest program that, with extra information y, prints x. A string x is incompressible relative to y if $C(x|y)$ is large in the appropriate sense. For details see [6]. Applying the *incompressibility method* one uses the fact that both absolutely and relative to any fixed string y, there are incompressible strings of every length, and that *most* strings are nearly incompressible, by *any* standard. [3] Another easy one is that significantly long subwords of an incompressible string are themselves nearly incompressible by *any* standard, even relative to the rest of the string. The argument invariably says that if a desired property does not hold, then in contrast to the assumption the object can be compressed. This yields the required contradiction. In the sequel we use the following easy facts (sometimes only implicitly).

Lemma 1 *Let c be a positive integer. For every fixed y, every finite set A contains at least $(1-2^{-c})|A|+1$ elements x with $C(x|A,y) \geq \lfloor \log |A| \rfloor - c$.*

Lemma 2 *If A is a set, then for every y every element $x \in A$ has complexity $C(x|A,y) \leq \log |A| + O(1)$.*

The first lemma is proved by simple counting. The second lemma holds since x can be described by first describing A in $O(1)$ bits and then giving the index of x in the enumeration order of A.

2 Shellsort

A Shellsort computation consists of a sequence of comparison and inversion (swapping) operations. In this analysis of the average-case lower bound we count just the total number of data movements (here inversions) executed. The same bound 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. We formulate the proof in the crisp format of incompressibility.

Theorem 1 *The average number of inversions in p-pass Shellsort on lists of n keys is at least $\Omega\left(pn^{1+1/p}\right)$ for every increment sequence. The average is taken with all lists of n items equally likely (uniform distribution).*

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 $\sum_{i=1}^{n} m_{i,k}$ is the number of inversions in

[2] The trivial lower bound is $\Omega(pn)$ comparisons since every element needs to be compared at least once in every pass.

[3] By a simple counting argument one can show that whereas some strings can be enormously compressed, like strings of the form $11\ldots1$, the majority of strings can hardly be compressed at all. For every n there are 2^n binary strings of length n, but only $\sum_{i=0}^{n-1} 2^i = 2^n - 1$ possible shorter descriptions. Therefore, there is at least one binary string x of length n such that $C(x) \geq n$. Similarly, for every length n and any binary string y, there is a binary string x of length n such that $C(x|y) \geq n$.

the initial permutation of pass k, and that the insertion sort in pass k requires precisely $\sum_{i=1}^{n}(m_{i,k}+1)$ comparisons. Let M denote the total number of inversions:

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

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

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 easily reconstruct the initial permutation of pass k. □

Let M as in (1) be a fixed number. There are $n!$ permutations of n elements. Let permutation π be an incompressible permutation having Kolmogorov complexity

$$C(\pi|n,A,P) \geq \log n! - \log n. \tag{2}$$

where P is the decoding program in the following discussion. There exist many such permutations by lemma 1. Clearly, there is a fixed program that on input A, n reconstructs π from the description of the $m_{i,k}$'s as in Claim 1. Therefore, the minimum length of the latter description, including a fixed program in $O(1)$ bits, must exceed the complexity of π:

$$C(m_{1,1},\ldots,m_{n,p}|n,A,P) + O(1) \geq C(\pi|n,A,P). \tag{3}$$

An M as defined by (1) such that every division of M in $m_{i,k}$'s contradicts (3) would be a lower bound on the number of inversions performed. How many divisions 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

$$D(M) :== \binom{M+np-1}{np-1} \tag{4}$$

distinct divisions of M into np ordered nonnegative integral summands $m_{i,k}$'s. Every division can be indicated by its index j in an enumeration of these divisions. This is both obvious and an application of lemma 2. Therefore, a description of M followed by a description of j effectively describes the $m_{i,k}$'s. Fix P as the program for the reference universal machine that reconstructs the ordered list of $m_{i,k}$'s from this description. The binary length of this two-part description must by definition exceed the Kolmogorov complexity of the described object.

A minor complication is that we cannot simply concatenate two binary description parts: the result is a binary string without delimiter to indicate where one substring ends and the other one begins. To overcome this problem we encode one of the substrings *self-delimitingly*: a binary string $x = x_1 \ldots x_n$ of n bits can be encoded by $1^n 0 x_1 \ldots x_n$. Iterating this process with, say $n = n_1 \ldots n_m$ is a string of m bits, one can encode x by $1^m 0 n_1 \ldots n_m x_1 \ldots x_n$. The latter self-delimiting encoding has length $n + 2\log n + 1$ and can be parsed by the reference universal machine.

Encoding the M part of the description self-delimitingly we obtain:

$$\log D(M) + \log M + 2\log\log M + 1 \geq C(m_{1,1},\ldots,m_{n,p}|n,A,P).$$

We know that $M \leq pn^2$ since every $m_{i,k} \leq n$. We can assume[4] $p < n$. Together with (2) and (3), we have

$$\log D(M) \geq \log n! - 4\log n - 2\log\log n - O(1). \tag{5}$$

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

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

[4]Otherwise we require at least n^2 comparisons.

[5]Use the following formula ([6], p. 10),

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

The second term in the right-hand side equals[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 $n \leq M \leq pn^2$,

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

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

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

for $n \to \infty$. Since the righthand-side of (5) is asymptotic to $n \log n$ for $n \to \infty$, this yields

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

That is, the running time of the algorithm is as stated in the theorem for every permutation π satisfying (2). By lemma 1 at least a $(1-1/n)$-fraction of all permutations π require that high complexity. Then the following is a lower bound on the expected number of inversions of the sorting procedure:

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

This gives us the theorem. □

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 [10]. 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 $\Theta(n^{5/3})$ [5]);
3. For $p=3$, Shellsort requires $\Omega(n^{4/3})$ inversions (the best known upper bound is $O(n^{23/15})$ in [3]);
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 $\Omega(n \log n)$ inversions. When we consider comparisons, this is of course the lower bound of average number of comparisons for every sorting algorithm.
6. In general, for $p = p(n) > \log n$, Shellsort requires $\Omega(n \cdot p(n))$ inversions (it requires that many comparisons anyway since every pass trivially makes n comparisons).

In [13] 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 [7] 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 $\Theta(n \log n)$?

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

4 Acknowledgements

We thank Don Knuth, Ian Munro, and Vaughan Pratt for discussions and references on Shellsort.

References

[1] H. Buhrman, T. Jiang, M. Li, and P. Vitányi, New applications of the incompressibility method, pp. 220–229 in *the Proceedings of ICALP'99*, LNCS 1644, Springer-Verlag, Berlin, 1999.

[2] J. Incerpi and R. Sedgewick, Improved upper bounds on Shellsort, *Journal of Computer and System Sciences*, 31(1985), 210–224.

[3] S. Janson and D.E. Knuth, Shellsort with three increments, *Random Struct. Alg.*, 10(1997), 125-142.

[4] A.N. Kolmogorov, Three approaches to the quantitative definition of information. *Problems Inform. Transmission*, 1:1(1965), 1–7.

[5] D.E. Knuth, *The Art of Computer Programming, Vol.3: Sorting and Searching*, Addison-Wesley, 1973 (1st Edition), 1998 (2nd Edition).

[6] M. Li and P.M.B. Vitányi, *An Introduction to Kolmogorov Complexity and its Applications*, Springer-Verlag, New York, 2nd Edition, 1997.

[7] T. Jiang, M. Li, and P. Vitányi, The average-case complexity of Shellsort, Preliminary version, pp. 453–462 in *the Proceedings of ICALP'99*, LNCS 1644, Springer-Verlag, Berlin, 1999.

[8] T. Jiang, M. Li, and P. Vitányi, The expected size of Heilbronn's triangles, *Proc. 14th IEEE Conference on Computational Complexity*, 1999, 105-113.

[9] A. Papernov and G. Stasevich, A method for information sorting in computer memories, *Problems Inform. Transmission*, 1:3(1965), 63–75.

[10] C.G. Plaxton, B. Poonen and T. Suel, Improved lower bounds for Shellsort, *Proc. 33rd IEEE Symp. Foundat. Comput. Sci.*, pp. 226–235, 1992.

[11] V.R. Pratt, *Shellsort and Sorting Networks*, Ph.D. Thesis, Stanford Univ., 1972.

[12] R. Sedgewick, Analysis of Shellsort and related algorithms, presented at the *Fourth Annual European Symposium on Algorithms*, Barcelona, September, 1996.

[13] R. Sedgewick, Open problems in the analysis of sorting and searching algorithms, Presented at *Workshop on Prob. Analysis of Algorithms*, Princeton, 1997.

[14] D.L. Shell, A high-speed sorting procedure, *Commun. ACM*, 2:7(1959), 30–32.

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