On the Average-Case Complexity of Shellsort

Paul Vitányi[*]
CWI and University of Amsterdam

Abstract

We prove a lower bound expressed in the increment sequence on the average-case complexity (number of inversions which is proportional to the running time) of Shellsort. This lower bound is sharp in every case where it could be checked. We obtain new results e.g. determining the average-case complexity precisely in the Yao-Janson-Knuth 3-pass case.

1 Introduction

The question of a tight general lower bound or upper bound on the average-case complexity of Shellsort (due to D.L. Shell [12]) has been open for more than five decades [5]. We use "average" throughout in the sense of "arithmetic mean," and the average-case complexity is the average-case of the number of inversions. (The number of inversions is proportions to the running time, and the number of comparisons in p-pass Shellsort of n keys is np larger than the number of inversions.) We present an average-case lower bound on the number of inversions for a p-pass Shellsort with increments $h_1, h_2, \ldots, h_p$ for every number of passes and increment sequences.

Shellsort sorts *in situ* a list of n elements in p passes using a sequence of increments $h_1, \ldots, h_p$ with $n > h_1 > \cdots > h_p$. In the kth pass the main list is divided in h_k separate sublists of length n/h_k, where the jth sublist consists of the elements in positions j mod h_k of the main list ($j = 1, \ldots, h_k$). 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 $h_1, \ldots, h_p$ satisfying $h_p = 1$ to ensure sortedness of the final list. Shellsort is used for example in the C standard library, in the uClibc library, Linux kernel, and bzip2 compressor [13].

[*]Address: CWI, Science Park 123, 1098 XG Amsterdam, The Netherlands. Email: `paulv@cwi.nl`

1.1 Previous Work

Let log denote the binary logarithm. The original $\log n$-pass increment sequence $\lfloor n/2 \rfloor, \lfloor n/4 \rfloor, \ldots, 1$ of Shell [12] uses worst case $\Theta(n^2)$ number of inversions, but Papernov and Stasevich [8] showed that another related increment sequence uses worst-case number of inversions $\Theta(n^{3/2})$ and Pratt [10] extended this to a class of all nearly geometric increment sequences and proved this bound was tight in the sense of coinciding with the upper bound. Incerpi and Sedgewick [2] constructed a family of $O(\log n)$-length increment sequences for which Shellsort runs in $O(n^{1+\epsilon/\sqrt{\log n}})$ number of inversions, for all $\epsilon > 0$. Plaxton, Poonen and Suel [9] proved an $\Omega(n^{1+\epsilon/\sqrt{p}})$ lower bound for p passes of Shellsort using any increment sequence, for every $\epsilon > 0$ and showed that this bound is tight for $p = \Omega(\log n)$. 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. The currently best asymptotic method was found by Pratt [10]. It uses all $\log^2 n$ increments of the form $2^i 3^j < \lfloor n/2 \rfloor$ to obtain number of inversions $O(n \log^2 n)$ in the worst case. Moreover, since every pass takes at least n steps, the average-case complexity using Pratt's increment sequence is $\Theta(n \log^2 n)$. Knuth [5] shows $\Theta(n^{5/3})$ for the average-case case of $p = 2$ passes and Yao [15] derives an expression for the average case for $p = 3$ that gives not (yet) a direct bound but was used by Janson and Knuth to derive an upper bound of $O(n^{23/15})$ on the average-case complexity of 3-pass Shellsort. In [7] Jiang , Li and Vitányi derived a general lower bound of $\Omega(pn^{1+1/p})$ on the average-case complexity of p-pass Shellsort. This lower bound shows that the only possibility of Shellsort to run on the average in $\Theta(n \log n)$ inversions is for the number of passes p to satisfy $p = \Theta(\log n)$. Apart from this, no nontrivial results were known for the average case until the results presented here. A more detailed history can be found in [5].

1.2 Present Work

We show a lower bound on the average number of inversions (which is the same order of magnitude as the running time and the number of comparisons) of Shellsort expressed in the increment sequence used (Theorem 1). The proof uses the fact that most permutations of n keys have high Kolmogorov complexity. Since the number of inversions in the Shellsort process is not easily amenable to analysis, we analyze a simpler process. This simpler process has at most as many inversions (the sum of the *minor sequence* in Definition 1) as the original process. Hence a lower bound on the number of inversions in the simpler process is a lower bound on the number of inversions of the original process. We show that the largest number of inversions of each key in the kth pass of the simpler process is less than h_{k-1}/h_k where $h_1, \ldots, h_p$ is the increment sequence and $h_0 = n$ (Claim 2). Subsequently it is shown using the high Kolmogorov complexity of the permutation that most keys in each pass have a number of inversions close to the maximum. This gives a lower bound on the total number of inversions of the simpler process (Claim 3) and hence a lower bound for the original process. This holds for the chosen single permutation. Since all permutations but for a vanishing fraction (with growing n) have this high Kolmogorov complexity, the lower bound on the total number of inversions holds for the average case of the original Shellsort process (Theorem 1). The lower bound seems to be tight since it coincides with the known bounds. For 2-pass Shellsort Knuth in [5] determined the average-case complexity and the new lower bound on the average complexity coincides (Corollary 1). For 3-pass Shellsort Knuth and Janson [3], building on the work of Yao [15], gave an upper bound on the average-case complexity and the new lower bound coincides (Corollary 1). This yields the new result that the average-case complexity of Shellsort for this increment sequence is now determined. They [3] conjecture an upper bound on the average-case complexity for another increment sequence. The lower bound on the average-case complexity established here for this sequence coincides with this upper bound (Corollary 1). For the logarithmic increment sequences of Shell [12], Papernov and Stasevich [8], Hibbard [1], and Pratt [10] also reported in [5], the lower bound on the average-case complexity for the respective increment sequences is $\Omega(n \log n)$ (Corollary 2). No upper bound on the average-case complexity is known for any of these increment sequences. For the square logarithmic increment sequence of Pratt [10] the average-case complexity is known. Again, the lower bound given here coincides (Corollary 3).

2 The Lower Bound

A Shellsort computation consists essentially of a sequence of inversion (swapping) operations. We count the total number of data movements (here inversions). Keys in the input permutation go by inversions to their final destination. The sequences of inversions constitute insertion paths. The proof is based on the following intuition. There are $n!$ different permutations of n keys. Given the sorting process (the insertion paths in the right order) one can recover the original permutation from the sorted list. The sum of the lengths of the insertion paths must be at least as large as the length of a most concise representation of the starting permutation.

Theorem 1. *The average-case number of inversions in a p-pass Shellsort algorithm on n keys with increment sequence $h_1, \ldots, h_p$, and denoting $h_0 = n$, has a lower bound of $\Omega\left(n \sum_{k=1}^{p} h_{k-1}/h_k\right)$.*

Proof. Let the list to be sorted consist of a permutation π of the keys $1, \ldots, n$. Let A be a p-pass Shellsort algorithm with increments $(h_1, \ldots, h_p)$ such that h_k is the increment in the kth pass and $h_p = 1$. Denote the original permutation by $\pi = \pi_0$ and the permutation resulting from pass k by π_k. In each permutation the keys are ordered left-to-right. In the final permutation $\pi_p = 12 \ldots n$ the least key 1 is on the left end and the greatest key n is on the right end.

For $k = 1, 2, \ldots, p$, the kth pass starts from π_{k-1} and this list (or permutation) is divided into h_k separate sublists or h_k*-chains* of length n/h_k, where the hth h_k-chain ($1 \leq h \leq h_k$) consists of the keys in positions $j \bmod h_k = h$ of the main list π_{k-1} ($j = 1, \ldots, n$). The insertion sort of a h_k-chain goes as follows. We start at the left end. If the second key is less than the first key then the second key is swapped with the first key. Otherwise nothing happens. This creates a new h_k-chain. If the third key is smaller than the first key or the second key in the new h_k-chain, then the third key is inserted in its correct position in the $<$-order before the first key or in between the first key and the second key. Otherwise nothing happens. We continue this way. The ith key is inserted in its correct position in the $<$-order in the initial segment of the current h_k-chain consisting of the first key through the $(i-1)$th key. All keys greater than the ith key in this initial segment move one position to the right. This is possible since the inserted key left a gap at the ith position of the current h_k-chain. An *inversion* is a swap of key i with key j which changes list $\ldots ij \ldots$ to list $\ldots ji \ldots$. We can view the insertion above as the ith key changing place with the key before it (an inversion), then changing place with the key before that (a second inversion), and so on, untill it ends up in its correct position. The inversions it had to make to do so is called its *insertion path.* By the time the final key is inserted in its correct position in the $<$-order the h_k-chain involved is sorted.

All keys $i = 1, 2, \ldots, n$ reside in an h_k-chain. Let $m_{i,k}$ be the number of inversions of key i in its h_k-chain in this sorting process. At the end of the sorting process the h_k-many h_k-chains are merged to establish permutation π_k by putting the jth key of the hth sorted h_k-chain into position $h+(j-1)h_k$ of permutation π_k ($1 \leq h \leq h_k$). That is, the jth position of a h_k-chain has the same position in permutation π_{k-1} as it has in permutation π_k. The sum

$$T = \sum_{i=1}^{n} \sum_{k=1}^{p} m_{i,k} \tag{1}$$

is the total number of inversions that algorithm A performs.

DEFINITION 1. Let n, π, the increment sequence $h_1, \ldots, h_p = 1$ and the Shellsort algorithm be as described above. A *simple Shellsort algorithm* is a Shellsort algorithm where the insertion sort of the h_k-chains is changed. For all keys $i = 1, 2, \ldots, n$ and passes $k = 1, 2, \ldots, p$, when it is the turn of key i in position j of its h_k-chain, it does the following. It moves a number $l_{i,k}$, $0 \leq l_{i,k} \leq n/h_k - j$, to the left by inversions subject to the following condition. The sequence $l_{1,1}, \ldots, l_{n,p}$ is such that pass 1 starts with π and pass p finishes with the sorted list of keys $12 \ldots n$. The *minor sequence* $S_n = n_{1,1}, \ldots, n_{n,p}$ is the lexicographic in i, k ($1 \leq i \leq n$ and $1 \leq k \leq p$) the first sequence of a simple Shellsort algorithm such that $T' = \sum_{i=1}^{n} \sum_{k=1}^{p} n_{i,k}$ is least over all $\sum_{i=1}^{n} \sum_{k=1}^{p} n_{i,k}$ for sequences $l_{1,1}, \ldots, l_{n,p}$ associated with simple Shellsort algorithms. Denote $T_i = \sum_{k=1}^{p} n_{i,k}$ for all $1 \leq i \leq n$ such that $T' = \sum_{i=1}^{n} T_i$.

CLAIM 1. *Given n, A and all $n_{i,k}$'s in appropriate fixed order, we can computably reconstruct the original permutation π.*

Proof. Let the sequence of permutations resulting from the minor sequence be $\pi = \rho_0, \rho_1, \ldots, \rho_p = 12 \ldots n$. The $n_{i,p}$'s in the appropriate order specify the initial permutation ρ_{p-1} of pass p. For $k = p, p-1, \ldots, 1$ given ρ_k we can in this way reconstruct the initial permutation of pass k. Hence given $\rho_p = 12 \ldots n$ we can reconstruct the original permutation $\rho_0 = \pi$ given the data items in the claim. $\square$

CLAIM 2. (i) $T' \leq T$.

(ii) $n_{i,k} < h_{k-1}/h_k$ for all $1 \leq i \leq n$, $1 \leq k \leq p$.

(iii) For every i ($1 \leq i \leq n$) holds that from $T_i = \sum_{k=1}^{p} n_{i,k}$ one can extract the $n_{i,k}$'s in the order $n_{i,1}, \ldots, n_{i,p}$.

Proof. (i) By Definition 1 each Shellsort sequence $m_{1,1}, \ldots, m_{n,p}$ is a simple Shellsort sequence. Hence by Definition 1 and the minimality of the minor sequence item (i) holds.

(ii) Let the sequence of permutations of the keys resulting from pass 1 through p using the $n_{i,k}$'s ($1 \leq i \leq n$, $1 \leq k \leq p$) be $\rho_1, \ldots, \rho_p = 12 \ldots n$, respectively. Assume by way of contradiction that there exist i, k ($1 \leq i \leq n$, $1 \leq k \leq p$) with $i + k$ least such that $n_{i,k} \geq h_{k-1}/h_k$. (In the following the rounding is ignored.) Define $n^{(1)}_{i,k-1} := n_{i,k-1} + 1$ and $n^{(1)}_{i,k} := n_{i,k} - h_{k-1}/h_k$, while $n^{(1)}_{j,h} = n_{j,h}$ otherwise. If $n^{(1)}_{i,k-1} \geq h_{k-2}/h_{k-1}$ then $n^{(2)}_{i,k-1} := n^{(1)}_{i,k-1} + 1$ and $n^{(2)}_{i,k-1} := n_{i,k-1} - h_{k-2}/h_{k-1}$, while $n^{(2)}_{j,h} = n^{(1)}_{j,h}$ otherwise. This process is repeated as often as necessary. However, it is not possible that it goes all the way to the 1st pass and results in $n^{(k-1)}_{i,1} \geq h_0/h_1$. Namely this would imply that key i moves to a position greater than n/h_1 in its h_1-chain of length n/h_1 which is impossible. Assume that the process above is repeated $l < k$ times. The resulting permutations of the passes are $\rho_1, \ldots, \rho_{k-l-1}\sigma_{k-l} \ldots \sigma_k \rho_{k-1} \ldots \rho_p$ where $\sigma_{l-k}, \ldots, \sigma_k$ are possibly new permutations of the n keys. However for $l > 0$ the $\sum_{i=1}^n \sum_{k=1}^p n^{(l)}_{i,k} < T'$, contradicting the minimality of the minor sequence $n_{1,1}, \ldots, n_{n,p}$.

(iii) Let $1 \leq i \leq n$. The number of inversions which move key i from its position as the jth key in π from the left end to its final position as the ith key from the left end in $\pi_p = 12 \ldots n$ is $T_i = \sum_{k=1}^p n_{i,k}$. The distance traveled is represented in a mixed radix system with radices $h_1, h_2, \ldots, h_p$. The question asked is whether this representation is unique or not. By the minimality of T' all $T_1, \ldots, T_p$ are minimal. Hence in each $n_{i,1}, \ldots, n_{i,p}$ the $n_{i,k}$'s are as great as possible subject to $n_{i,k} < h_{k-1}/h_k$ (item (ii)) and $T_i = \sum_{k=1}^p n_{i,k} h_k$ in the order $k = 1, \ldots, p$, for each $1 \leq i \leq n$. Since $h_0 > h_1 > \cdots > h_p$ this shows the representation is unique. $\square$

There are $n!$ permutations π of n keys to be sorted. Below we use the plain Kolmogorov complexity defined by Kolmogorov in [4] and denoted by C in the text [6]. Let x, y, z be natural numbers. Here is used that $C(x|y) \leq \log x + O(1)$ for all x and y and $C(z|y) \geq \log x - c$ for fixed y and $|bin(x)| - |bin(x)|/2^c$ numbers $z \neq x$ such that $|bin(z)| = |bin(x)|$. Here $bin(x)$ is the binary representation of x and $|bin(x)|$ is its length in bits. Iterating the use of standard pairing functions, such natural numbers may consist of fixed finite sequences of natural numbers.

There are $n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$ permutations of n keys (here π is for once the number π). By Stirling's approximation used below we are justified to choose the permutation π with Kolmogorov complexity

$$C(\pi|n, A, P) \geq n \log n - 3n, \tag{2}$$

with A the algorithm used in this p-pass Shellsort (including the increment sequence), and P a constant-size algorithm to process all the information and to output π.

Denote the minor sequence $n_{1,1}, \ldots, n_{n,p}$ by $S_n = T_1, T_2, \ldots, T_p$. A computable description of S_n, given n, A and Q (an $O(1)$ bit program included in P above), requires at most

$$|descr(S_n)| = (\sum_{i=1}^{n} \sum_{k=1}^{p} \log n_{i,k}) + D \tag{3}$$

bits where D is the number of bits required to be able to parse the main part of $descr(S_n)$ into its constituent parts, that is, the concatenated bit strings $n_{i,k}$ of length $\log n_{i,k}$ $(1 \leq i \leq n,\ 1 \leq k \leq p)$. By Claim 1 we can compute permutation π from $descr(S_n)$, given n, A and Q. Hence

$$|descr(S_n)| \geq C(\pi | n, A, P). \tag{4}$$

From (2) and (4) it follows that

$$|descr(S_n)| \geq n \log(n/8). \tag{5}$$

Claim 3. Writing $h_0 = n$ we have

$$\sum_{i=1}^{n} \sum_{k=1}^{p} n_{i,k} = \Omega \left(n \sum_{k=1}^{p} h_{k-1}/h_k \right).$$

Proof. In pass 1 for every key i $(1 \leq i \leq n)$ we have $n_{i,1} < n/h_1$. In general by Claim 2 item (ii) for every pass k $(1 \leq k \leq p)$ $n_{i,k} < h_{k-1}/h_k$. Since $\prod_{k=1}^{p} h_{k-1}/h_k = h_0/h_p = n$ we have by (3) and (5) that

$$\begin{aligned} \frac{|descr(S_n)|}{n} &= \frac{\sum_{k=1}^{p} n(\log(h_{k-1}/h_k) - a_k)}{n} + \frac{D}{n} \\ &= \log n - \sum_{k=1}^{p} a_k + \frac{D}{n} \geq \log \frac{n}{8}, \end{aligned} \tag{6}$$

where $a_k = \log(h_{k-1}/h_k) - 1/n \sum_{i=1}^{n} \log n_{i,k}$ for $k = 1, 2, \ldots, p$ and $a_k \geq 0$ by Claim 2 item (ii).

We show that $D/n = o(\log n)$. To be able to parse $descr(S_n)$ into its constituent descriptions $descr(T_1), \ldots, descr(T_n)$ it suffices that $D = \sum_{i=1}^{n} 2 \log |descr(T_i)| + O(1)$. Since $T_i = \sum_{k=1}^{p} n_{i,k} \leq n$ we have $D \leq 2n \log\log n + O(1)$ $(1 \leq i \leq n)$. Namely, to encode every part T_i for $1 \leq i \leq n$ such that it can be parsed from the total we double each bit of the part except the last bit which is followed by its complement. This doubles the length of each part. Additionally we require $O(1)$ bits for a program to retrieve the T_i's and extract $n_{i,1}, \ldots, n_{i,p}$ from them. This can be done in a unique way by Claim 2 item (iii). The total of the description D is $o(n \log n)$ bits.

Hence up to lower order terms the last inequality of (6) is rewritten as $\sum_{k=1}^{p} a_k \leq 3$. Since $a_k \geq 0$ for every $1 \leq k \leq p$ we have $a_k \leq 3$. Writing a_k out and reordering this gives up to lower order terms

$$\log(h_{k-1}/h_k) \leq 1/n \sum_{i=1}^{n} \log n_{i,k} + 3,$$

and by exponentiation of both sides of the inequality one obtains

$$h_{k-1}/h_k = O((\prod_{i=1}^{n} n_{i,k})^{1/n}).$$

By the inequality of the arithmetic and geometric means and rearranging we obtain $\sum_{i=1}^{n} n_{i,k} = \Omega(nh_{k-1}/h_k)$ for every $1 \leq k \leq p$. Therefore, $\sum_{k=1}^{p} \sum_{i=1}^{n} n_{i,k} = \Omega(n \sum_{k=1}^{p} h_{k-1}/h_k)$. □

Since $T' \leq T$ by Claim 2 item (i), a lower bound for T' is also a lower bound for T. Therefore Claim 3 proves the statement of the theorem for the particular permutation π.

By Stirling's approximation $\log n! \sim n \log(n/e) + \frac{1}{2} \log n + O(1) \approx n \log n - 1.44n + \frac{1}{2} \log n + O(1)$. Therefore $n \log n - 1.5n \leq \log n! \leq n \log n - n$ for large n. Hence by [6, Theorem 2.2.1] which uses a simple counting argument, at least a $(1 - 1/n)$-fraction of all permutations π on n keys satisfy (2). Since $1/n \to 0$ for $n \to \infty$, and for all permutations π on n keys we have $T = O(n^2)$, a $(1 - 1/n)$th fraction of all $n!$ permutations has a lower bound as permutation π does, and a $1/n$th fraction has a lower bound of at least $\Omega(0)$ and at most $\Omega(n^2)$. Hence the lower bound on the average number of inversions. □

Corollary 1. For $p = 2$ with $h_1 = n^{1/3}$, and $h_2 = 1$ this yields

$$T = \Omega(n(n^{1-1/3} + n^{1/3}) = \Omega(n^{5/3}),$$

which coincides with the best number of inversions for 2-pass Shellsort $T = \Theta(n^{5/3})$ using the same increment sequence $h_1 = n^{1/3}, h_2 = 1$ as given by [5].

For $p = 3$ with $h_1 = n^{7/15}$, $h_2 = n^{1/5}$, and $h_3 = 1$ this yields

$$T = \Omega(n(n^{1-7/15} + n^{7/15-1/5} + n^{1/5}) = \Omega(n^{1+8/15}) = \Omega(n^{23/15}).$$

The upper bound of $O(n^{23/15})$ for 3-pass Shellsort using the same increment sequence $h_1 = \Theta(n^{7/15}), h_2 = \Theta(n^{1/5}), h_3 = 1$ with the additional restriction that $gcd(h_1, h_2) = 1$ is given in [3]. This reference uses a complicated probabilistic analysis based on the still more complicated combinatorial characterization in [15]. Together with the lower bound we establish the new fact that the average number of inversions of 3-pass Shellsort with this increment sequence is $\Theta(n^{23/15})$.

In Section 10 of [3] it is conjectured that with $h_1 \approx n^{1/2}$ and $h_2 \approx n^{1/4}$ ($h_3 = 1$) one may obtain an average-case number of inversions of $O(n^{3/2})$. Using the theorem above shows that $T = \Omega(n(n^{1-1/2} + n^{1/2-1/4} + n^{1/4}) = \Omega(n^{3/2})$. Therefore, if the conjecture on the upper bound is true then 3-pass Shellsort has an average-case number of inversions of $\Theta(n^{3/2})$ for this increment sequence.

Corollary 2. The increment sequence $h_1, \ldots, h_p$ with $p = \lfloor \log n \rfloor$ of Papernov and Stasevich in [8] is $h_1 = n/2 + 1, h_2 = n/2^2 + 1, \ldots, h_p = n/2^{\lfloor \log n \rfloor} + 1$. The worst-case number of inversions reported by [8] is $\Theta(n^{3/2})$. Since $h_{k-1}/h_k \approx 2$ and $2 = \Omega(1)$, the theorem above gives a lower bound on the average number of inversions of $T = \Omega(n \sum_{k=1}^{\Theta(\log n)} \Omega(1)) = \Omega(n \log n)$.

The increment sequence of Hibbard [1] with increment sequence $2^k - 1$ until it passes n has a worst-case number of inversions $\Theta(n^{3/2})$. With a similar analysis as before it has a lower bound on the average-case of $T = \Omega(n \log n)$. It is conjectured to lead to an average-case number of inversions of $O(n^{5/4})$ in [14] reported in [5]. This conjecture is difficult to settle empirically. For $n = 100,000$ we have $\log n \approx n^{1/4}$. Hence we need to do many experiments for n much larger than $100,000$ to obtain evidence. The upper bound may well be $O(n \log n)$.

Pratt's logarithmic increment sequence (one of his "hypergeometric" sequences) in [10] also reported by [5] is $h_1, \ldots, h_p$ with $h_k = (3^k - 1)/2$ not greater than $\lceil n \rceil$. This increment sequence leads to a worst-case number of inversions of $\Theta(n^{3/2})$. In this case $h_{k-1}/h_k \approx 3$ and $3 = \Omega(1)$ and the number of passes is $p = \log_3 n$. The theorem above gives a lower bound on the average number of inversions of $T = \Omega(n \sum_{k=1}^{\Theta(\log n)} \Omega(1)) = \Omega(n \log n)$.

The original increment sequence used by Shell [12] was $\lfloor n/2 \rfloor, \lfloor n/2^2 \rfloor$, and so on for $\log n$ passes. Knuth [5] remarks that this is undesirable when the binary representation of n contains a long string of zeroes. It has the result that Shellsort runs in worst-case time $\Theta(n^2)$. Since this increment sequence satisfies the same analysis as the above one of [8] the lower bound on the average number of inversions is $\Omega(n \log n)$.

By [7] the average number of inversions of Shellsort can be $\Theta(n \log n)$ only for an increment sequence $h_1, \ldots, h_p$ with $p = \Theta(\log n)$. We have shown here that the lower bound on the average inversions is $\Omega(n \log n)$ for many increment sequences of this length. It is an open problem whether it can be proved that for some such increment sequence the average number of inversions is $O(n \log n)$.

COROLLARY 3. For Pratt's square logarithmic increment sequence $h_1, \ldots, h_p$ with $p = \Theta((\log n)^2)$, the average-case number of inversions is lower bounded by $T = \Omega(n \sum_{k=1}^{\Theta((\log n)^2)} \Omega(1)) = \Omega(n(\log n)^2)$. The precise average-case number (and worst-case number) of inversions is $\Theta(n(\log n)^2)$ in [10], and therefore the lower bound is tight.

3 Conclusion

The lower bound on the average case number of inversions of Shellsort using p passes in [7] is for worst-case increment sequences. Here we gave a lower bound on the average-case complexity for each increment sequence separately. This is in several cases larger than the lower bound of above reference. In fact, the lower bound given here seems to be tight as follows from the corollaries. If the increment sequence is the best possible for a given number of passes then the lower bound should reflect this. A tantalizing prospect is to obtain from the given lower bound one which is expressed only in the number n of keys to be sorted and the number of passes in the sorting process, and which is tighter than the lower bound of [7].

Acknowledgment

I thank Ronald de Wolf for comments on an early draft of this paper.

References

[1] T.N. Hibbard, An Empirical Study of Minimal Storage Sorting, *Commun. ACM*, 6:5(1963), 206-213.

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

[3] S. Janson, D. Knuth, Shellsort with three increments, *Random Structures Algorithms*, 10:1-2(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, 3rd Edition, 2008.

[7] T. Jiang, M. Li, P.M.B. Vitányi, A lower bound on the average-case complexity of Shellsort, *J. Assoc. Comp. Mach.*, 47:5(2000), 905–911.

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

[9] C.G. Plaxton, B. Poonen and T. Suel, Improved lower bounds for Shellsort, *FOCS'92*, pp. 226–235, 1992.

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

[11] R. Sedgewick, Open problems in the analysis of sorting and searching algorithms, Talk in Workshop on the probabilistic analysis of algorithms, Princeton, May, 1997.

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

[13] Shellsort in Wikipedia in December 2014, http://en.wikipedia.org/wiki/Shellsort, 2014.

[14] M.A. Weiss, Empirical study of the expected running time of Shellsort, *Comput. J.*, 34(1991), 88–91

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