KOLMOGOROV COMPLEXITY AND LOWER BOUNDS

Wolfgang J. Paul [+], [*]

1. Introduction

Several nonlinear lower bounds for the runtime of certain classes of sorting algorithms are known [1, 6] . They all apply to algorithms which treat the numbers to be sorted as something "atomic", i.e. whole numbers can be compared or transported, but "no arithmetic on the numbers" is performed, i.e. it is for instance not allowed to break the binary representation of numbers into parts and to perform computations with the fragments of the numbers. On the other hand at least two fast sorting algorithms make use of exactly such methods [1, 4] . In this paper we exhibit a class of sorting algorithms where the restriction of "no arithmetic on numbers" is partially removed and prove a nonlinear lower bound for the runtime of these algorithms.

For natural numbers i let $\text{bin}(i)$ denote the binary representation of i. We will consider the following sorting problem to be solved by multitape Turing machines:
Input: strings of the form $\text{bin}(i_1) \# \text{bin}(i_2) \# \ldots \# \text{bin}(i_m)$.
Output: $\text{bin}(j_1) \# \ldots \# \text{bin}(j_m)$ where
$(j_1,\ldots,j_m)$ is a permutation of $(i_1,\ldots,i_m)$
and $j_1 \leq \ldots \leq j_m$.

For strings w we denote by $|w|$ the length of w . A Turing machine is $t(n)$-time bounded if for all inputs of length n the machine stops after at most $t(n)$ steps. It easily follows from [6] that any machine which solves the sorting problem by rearranging the strings $\text{bin}(i_j)$ is $\Omega(n \log n)$-time bounded .

Also if any $o(n \log n)$-time bounded Turing machine solves this problem, then apparently at some point of the computation it must pull the sorted sequence in a magic way out of a black hat. Below we will make the notion of magic precise (as making numbers appear from a place on a tape where they apparently are not) and prove that without magic $\Omega(n \log n)$ steps are required for sorting.

2. Kolmogorov-complexity

The following definitions are basically due to Kolmogorov [5] .

Fix an encoding of 1-tape Turing machines into $\{0,1\}^*$ such that no encoding of one machine is the prefix of an encoding of some other machine. An easy way to achieve this

[+] Fakultät für Mathematik, Universität, D-4800 Bielefeld

[*] research partially supported by DAAD-grant 311-f-HSLA.

is the following. Let M be a machine, u the usual encoding and let v be the image of $bin(|u|)$ under the homomorphism which maps 0 to 00 and 1 to 10. Encode M into $v\,11\,u$. Denote by M_x a machine with encoding x. Let U be a universal Turing machine which given an input z first determines x and v such that $z = xv$, where x is an encoding of a Turing machine; then U simulates M_x an input v.

For $w \in \{0,1\}^*$ let

$$C(w) = \min\{|z| : U \text{ given } z \text{ prints } w\}$$

z is called the complexity of w. Clearly $C(w) \leq |w| + O(1)$ (take $z = xw$ where x is the encoding of a machine which does nothing). Also for all n there exist (many) w such that $C(w) \geq |w|$; this is established by a simple counting argument. A sequence $(w^i | i \in \mathbb{N}, w^i \in \{0,1\}^i)$ is called random if $C(w^i) = \Omega(i)$. We illustrate these concepts by proving a theorem due to Hennie [3].

Fact:

If M is a 1-tape Turing machine which accepts the language $\{w \#^{|w|} w \mid w \in \{0,1\}^*\}$ then M is $\Omega(n^2)$-time bounded[+].

Proof:

If $w \neq w'$, then no crossing sequence generated in the middle third of $w \#^{|w|} w$ equals a crossing sequence in the middle third of $w' \#^{|w'|} w'$ by the usual argument.

Let Σ be the alphabet of M. Fix some 1-1 homomorphism h from Σ^* to $\{0,1\}^*$. Let M' be a machine which given an input v enumerates all $w \in \{0,1\}^*$, simulates M an input $w \#^{|w|} w$ and tests if $h^{-1}(v)$ occurs as a crossing sequence. When that happens M' outputs w. Let x be the encoding of M' and CS any crossing sequence generated by M on input $w \#^{|w|} w$, then U given $xh(CS)$ prints w. Hence $C(w) \leq |x| + O(|CS|)$, or $|CS| = \Omega(C(w))$. Thus if $(w^i | i \in \mathbb{N})$ is a random sequence, then for any w^i all the $i/3$ crossing sequences generated in the middle third of $w^i \#^{|w^i|} w^i$ have length $\Omega(i)$. □

For $u, v \in \{0,1\}^*$ let

$$C(u|v) = \min\{|z| : U \text{ given } z \# v \text{ prints } u\} \text{ and}$$

$$I(u|v) = |u| - C(u|v).$$

[+] $\#^i$ denotes the string $\# \ldots \#$ (i-times)

$C(u|v)$ is called the complexity of u given v ,
$I(u|v)$ is called the information about u in v .

3. The complexity of sorting without magic

Let $\ell(n)$ be an easily computable function in the sense that $\mathrm{bin}(\ell(n))$ can be computed from $\#^n$ in $O(n)$ steps by some 2-tape Turing machine. Moreover let

$$n^{1/2} \log n \leq \ell(n) \leq n^{\alpha} \text{ , for some } \alpha < 1 \text{ .}$$

Let $b(n) = \sqrt{n/\ell(n)}$. Then

$$\ell(n) = \omega(b^2(n) \log n)^{+} \tag{3.1}$$

$$\log b(n) = \Omega(\log n) \text{ .} \tag{3.2}$$

Rather than studying sorting directly we study machines which solve the following "$\ell(n)$-matrix transposition problem" :

Input: $w \in \{0, 1\}^*$

Output: if $b(n)$ is not an integer then output w ;
if $b(n)$ is an integer and

$w = w_{11} \cdots w_{1b(n)} \cdots w_{b(n)1} \cdots w_{b(n)b(n)}$, $w_{ij} \in \{0, 1\}^{\ell(n)}$,

then output $w_{11}\, w_{21} \cdots w_{b(n)1} \cdots w_{1b(n)}\, w_{2b(n)} \cdots w_{b(n)b(n)}$

on the same squares where the input was.
(Transposition of a matrix which is stored row by row).

Clearly any sorting algorithm can be turned into a not much slower transposition algorithm:
For all i and j replace w_{ij} by $\mathrm{bin}(i + (j-1)\, b(n))\, w_{ij}$; separate the binary numbers by markers; sort; remove the markers and the additional leading bits [4] .
In order to keep the following calculations simple we consider only 1-tape k-head machines with alphabet $\{0, 1\}$. Thus let M be such a machine which is $t(n)$-time bounded and which performs matrix transposition. In what follows we write b for $b(n)$ and ℓ for $\ell(n)$. For inputs of length n divide the tape into blocks of

---

+ $f(n) = \omega(g(n))$ if $g(n) = o(f(n))$

$b \cdot \ell$ cells each ($b \cdot \ell$ is the length of a row of the matrix), such that the input occupies b consecutive blocks. Number the blocks with the integers such that the input occupies blocks 1 to b. Divide the time M spends into time intervals of $\ell\, b$ consecutive steps each. Number them from 1 to $T \le \lceil t(n)/(\ell\, b)\rceil$.
Add a time interval 0 which corresponds to the time before the computation. Let $B_i(t)$ denote the content of the block with number i at the end of the interval t. Let $I(t)$ be the set of the numbers of the blocks visited in time interval t. Then $I(t) \le 2k$ for all t.

We say M performs <u>magic</u> if for some $w \in \{w_{11},\ldots,w_{bb}\}$ and some $t \in \{1,\ldots,T\}$

$$\sum_{i\in I(t)} I(w|B_i(t)) \ge \sum_{i\in I(t)} I(w|B_i(t-1)) + \ell/b ,$$

that means after time interval t there is a lot more information about w in the single blocks which were visited than there was before (observe $\ell/b > n^{1/4}$).

<u>Theorem:</u>

If M does not perform magic then $t(n) \ge c \cdot n \log n$ for almost all n such that $b(n)$ is an integer and some positive constant c.

The <u>proof</u> follows the pattern of [6]. We assume that all heads are originally on the first bit of the input. Thus M can reach only blocks B_i with $|i| \le T$. For $|i| \le T$, $j \in \{1,\ldots,b\}$ and $t \le T$ let

$$A_{ij}(t) = \sum_{w\in B_j(T)} I(w|B_i(t))/\ell$$

that is intuitively the "number" of words w "present" in $B_i(t)$ which will finally be in block j. Of course there may be half words, quarter words etc. present.

<u>Lemma 1:</u>

For almost all n such that b is an integer there exist (many) inputs $W = w_{11}\ldots w_{bb}$ such that (3.3) and (3.4) hold:

$$A_{ij}(0) \le 2 \quad \text{for all } i \text{ and } j \tag{3.3}$$

$$\sum_{w\in\{w_{11},\ldots,w_{bb}\}} I(w|B) \le \ell\,(b+1) \quad \text{for all } B \in \{0,1\}^{\ell b} . \tag{3.4}$$

Proof:

We estimate the number of inputs W such that (3.3) or (3.4) does not hold. If (3.3) does not hold for W then for some i and j

$$\sum_{w \in B_j(T)} I(w|B_i(0)) > 2\ell \quad .$$

Hence

$$\sum_{w \in B_j(T)} (\ell - C(w|B_i(0)) > 2\ell$$

or

$$\sum_{w \in B_j(T)} C(w|B_i(0)) < \ell(b-2) \quad .$$

There are $b+1$ ways to choose i and b ways to choose j such that (3.3) does not hold because $A_{ij}(0) > 2$. If (3.3) does not hold for i and j then there is a sequence of strings $z = (z_1,\ldots,z_b)$ with $\Sigma |z_i| \leq \ell(b-2)$ such that w_{pj} = output of U given $z_p \# w_{i1} \ldots w_{ib}$ for all $p \in \{1,\ldots b\}$.
There are at most

$$2^{\ell(b-2)} (\ell b)^b$$

such sequences z. For each i, j and z there are $2^{\ell b(b-1)}$ ways to specify the words w_{pq}, $q \neq j$ i.e. the words which will finally be in blocks with numbers $\neq j$. If $i \neq b+1$ there are moreover 2^{ℓ} ways to specify w_{ij}. Thus the total number of inputs W such that (3.3) does not hold is bounded by

$$(b+1)\, b\, 2^{\ell(b-2)} + b\log\ell b \cdot 2^{\ell b^2 - \ell b + \ell}$$

$$\leq 2^{n-\ell} + b\log n + \log b(b+1) = o(2^n) \quad \text{by} \quad (3.1) \quad .$$

If (3.4) does not hold for W, then for some $B \in \{0,1\}^{\ell b}$:

$$\sum_{w \in \{w_{11},\ldots,w_{bb}\}} I(w|B) > \ell(b+1) \quad .$$

Hence

$$\sum_{w} (\ell - C(w|B)) > \ell(b+1)$$

or

$$\sum_{w} C(w|B) < b^2 \ell - \ell (b + 1) .$$

There are $2^{b\ell}$ ways to choose B. There are at most

$$2^{n - \ell(b+1)} \, n^{b^2}$$ sequences of strings

$z = (z_{11},\ldots,z_{bb})$ such that $\sum |z_{ij}| \leq b^2\ell - \ell(b + 1)$.

B together with the sequence of strings z specifies all w_{ij} namely w_{ij} = output of U given $z_{ij}\#B$. Thus the number of inputs W that do not satisfy (3.4) is at most

$$2^{b\ell} \cdot 2^{n - \ell(b+1) + b^2 \log n}$$

$$= 2^{n - b + b^2 \log n} = o(2^n) \quad \text{by (3.1)} . \qquad \square$$

Choose input W such that (3.3) and (3.4) hold.

For $t \leq T$ let

$$K(t) = \sum_{i=-T}^{T} \sum_{j=1}^{b} A_{ij}(t) \log A_{ij}(t) .$$

Then $A_{ij}(0) \leq 2$ for all i and j. Thus $K(0) \leq 2(2T + 1)\, b$.

On the other hand let M' be a machine which given input $\mathrm{bin}(i) \# \mathrm{bin}(j) \# v_1 \ldots v_n$ $(v_i \in \{0, 1\})$ prints $v_i \ldots v_j$. Let x be an encoding of M'. Then U given input $x\, \mathrm{bin}((i - 1)\ell) \# \mathrm{bin}(i\ell) \# B_j(T)$ prints w_{ji}. Thus

$$C(w|B_j(T)) \leq 2 \log n + O(1) \quad \text{for all } w \in B_j(T) .$$

Thus for all $w \in B_j(T)$:

$$C(w|B_j(T)) \leq 2 \log n + O(1) \quad \text{or}$$

$$I(w|B_j(T)) \geq \ell - 2 \log n - O(1) .$$

Thus

$$A_{jj}(T) \geq b\left(1 - \frac{2 \log n + O(1)}{\ell(n)}\right) = \Omega(b) \quad .$$

Hence

$$K(T) \geq \sum_j A_{jj}(T) \log A_{jj}(T)$$

$$\geq b\,\Omega(b) \log \Omega(b)$$

$$= \Omega(b^2 \log n) \qquad \text{by} \quad (3.2) \quad .$$

We prove

Lemma 2: $K(t) - K(t-1) \leq O(b)$ for all $t \in \{1,\ldots,T\}$.

This implies of course

$$T \geq (K(T) - K(0)) / O(b)$$

$$= \Omega((b^2 \log n - (4T+2)\, b)/b)$$

$$\geq D(b \log n - (4T+2)) \quad \text{for some } D \text{ and large } n.$$

Hence $T(1 + 4D) \geq Db \log n$, and the theorem follows.
It remains to prove Lemma 2 :

$$K(t) - K(t-1) = \sum_j \sum_{i \in I(t)} (A_{ij}(t) \log A_{ij}(t) - A_{ij}(t-1) \log A_{ij}(t-1))$$

$$\leq \sum_j \Big[\Big(\sum_{\substack{i \in I(t) \\ A_{ij}(t) \geq 1}} A_{ij}(t)\Big) \log \Big(\sum_{\substack{i \in I(t) \\ A_{ij}(t) \geq 1}} A_{ij}(t)\Big)$$

$$- \Big(\sum_{i \in I(t)} A_{ij}(t-1)\Big) \log \Big(\frac{1}{2k} \sum_{i \in I(t)} A_{ij}(t-1)\Big)\Big]$$

because the function $f(x) = x \log x$ $(f(0) := 0)$ is negative in $[0, 1]$, monotone in $[1, \infty)$ and convex in $[0, \infty)$.

Because no magic is performed the following is true for all j and all $w \in B_j(T)$

$$\sum_{i \in I(t)} I(w|B_i(t)) \leq \sum_{i \in I(t)} I(w|B_i(t-1)) + \ell/b \quad .$$

Thus

$$\sum_{\substack{i\in I(t)\\ A_{ij}\geq 1}} A_{ij}(t) \leq \sum_{i\in I(t)} A_{ij}(t)$$

$$= \frac{1}{\ell} \sum_{w\in B_j(T)} \sum_{i\in I(t)} I\,(w|B_i(t))$$

$$\leq \frac{1}{\ell} \sum_{w\in B_j(T)} \Big(\sum_{i\in I(t)} I\,(w|B_i(t-1)) + \ell/b \Big)$$

$$= \sum_{i\in I(t)} A_{ij}(t-1) + 1 \quad .$$

Therefore $K(t) - K(t-1) \leq \sum_j S(j)$ where

$$S(j) = \Big(\sum_{i\in I(t)} A_{ij}(t-1) + 1 \Big) \log \Big(\sum_{i\in I(t)} A_{ij}(t-1) + 1 \Big)$$

$$- \sum_{i\in I(t)} A_{ij}(t-1) \log \Big(\frac{1}{2k} \sum_{i\in I(t)} A_{ij}(t-1) \Big) \quad .$$

Now for all j such that

$$\sum_{i\in I(t)} A_{ij}(t-1) < 1 \tag{3.5}$$

we find $S(j) \leq 0\,(1)$ as $f(x) = x \log x$ is bounded in $[0, 1]$.

Using $\log (x+1) \leq \log x + \frac{1}{x}$ we find for all j with

$$\sum_{i\in I(t)} A_{ij}(t-1) \geq 1 \ : \tag{3.6}$$

$$\Big(\sum_{i\in I(t)} A_{ij}(t-1) + 1 \Big) \log \Big(\sum_{i\in I(t)} A_{ij}(t-1) + 1 \Big)$$

$$\leq \sum_{i\in I(t)} A_{ij}(t-1) \Big(\log \Big(\sum_{i\in I(t)} A_{ij}(t-1) + 1 \Big)$$

$$+ \sum_{i\in I(t)} A_{ij}(t-1) + 1$$

thus

$$S(j) \leq 1 + (2 + \log(2k)) \sum_{i \in I(t)} A_{ij}(t-1) .$$

Finally for some positive constant E

$$\begin{aligned} K(t) - K(t-1) &\leq \sum_{(3.5)} S(j) + \sum_{(3.6)} S(j) \\ &\leq 0(b) + E \sum_{j=1}^{b} \sum_{i \in I(t)} A_{ij}(t-1) \\ &= 0(b) + E \sum_{i \in I(t)} \sum_{j=1}^{b} \sum_{w \in B_j(T)} I(w|B_i(t))/\ell \\ &\leq 0(b) + \frac{E}{\ell} \sum_{i \in I(t)} \sum_{w \in \{w_{11}, \ldots, w_{bb}\}} I(w|B_i(t)) \\ &\leq 0(b) + \frac{E}{\ell} \sum_{i \in I(t)} \ell(b+1) \qquad \text{by} \quad (3.4) \\ &= 0(b) + 2kE(b+1) = 0(b) . \end{aligned}$$

This proves Lemma 2 and the theorem. □ □

We conclude by indicating how magic can be performed for two numbers
$u = u_1 \ldots u_\ell$ and $v = v_1 \ldots v_\ell$ $(u_i, v_i \in \{0, 1\})$: Form
$w = w_1 \ldots w_\ell$ by $w_i = u_i + v_i \mod 2$ $(1 \leq i \leq \ell)$.

If u and v are appropriately chosen, then
$I(u|w)$ and $I(u|v)$ are both close to 0 , however $I(u|wv)$ is close to ℓ .

4. References

[1] A. Aho, J. Hopcroft, J. Ullman: The design and analysis of computer algorithms, Addison-Wesley, 1974.

[2] G. Chaitin: A theory of program size formally identical to information theory, IBM Yorktown, Heights, preprint, 1974.

[3] F. Hennie: One-tape off-line Turing machine computations, Information and Control, 8, 553-578, 1965.

[4] J. Hopcroft, W. Paul, L Valiant: On time versus space and related problems, 16th IEEE-FOCS, 57-64, 1975.

[5] A. Kolmogorov: Three approaches to the quantitive definition of information, Problems of Information Transmission, 1, 1-7, 1965.

[6] H.J. Stoß: Rangierkomplexität von Permutationen, Acta Informatica, 2, 80-96, 1973.