"Ecologic" Computations

B. Durand, N. K. Vereshchagin, M. A. Ushakov

September 8, 2002

1 Introduction

Most of us are familiar with the situation when there is no space on hard disk to save a file. We then clean up the space with the risk of deleting some data that could be important in the future. Today's fast growth of disk sizes solves this problem to some extent, but look at the problem in a wider scale: to produce a new disk at the factory we need to destroy information that could be needed in the future. Indeed, the factory producing hard disks spoils irreversibly the environment. Assume that the entire environment (including all the world's hard disks) is the input data to the computation and that we do not have other space to perform the computation. We call such computations *ecological*.

In this paper, we study the power of such computations in multi-tape Turing machines computational model. In other words, the input data are the initial contents of all tapes of the machine. For simplicity, we consider only decision problems, i.e. functions with values 0,1.

Let $\mathbb{B}$ denote $\{0,1\}$, $\mathbb{B}^n$ the set of all binary strings of length n, $\mathbb{B}^*$ the set of all finite binary strings, Ω the set of all infinite binary sequences, $\Psi = \mathbb{B}^{\mathbb{Z}}$ the set of all bi-infinite binary sequences.

We will consider total and partial functions

$$f : (\mathbb{B}^*)^r \times \Omega^s \times \Psi^t \to \mathbb{B}, \tag{1}$$

where $r = 0, 1$ and $s + t > 0$, that is, functions that have at least one infinite argument and at most one finite one. The usual notion of computability of such functions is defined in terms of oracle Turing machines. An oracle Turing machines has one work tape that initially contains the input string if $r = 1$ or is empty if $r = 0$. Besides, the machine has an extra oracle tape where it can write a question to the oracle, that is, the number $i \leq s + t$ of the infinite argument and the number of its character which the machine wants to know. The specified character is then immediately available on the oracle tape.

The same notion of computability may be defined using Turing machines without oracle as follows (we will need in the sequel this model).

Definition 1. Consider machines having $r + s + t$ read only tapes that initially contain the input data (for writing bi-infinite inputs we will use bi-infinite tapes). The machine has one extra working tape. The machine has two final states, s_0 and s_1. If the machine gets into the s_0 (s_1) state, it halts with the result 0 (1).

We say that a partial function (1) is computable by a machine if for all inputs x from the domain of f the output of the machine is equal to $f(x)$.

Let us define now several notions of "ecological" computability of functions of the type (1).

Definition 2. An *ecological* Turing machine is a Turing machine with $r + s + t$ tapes that initially contain the input data. The machine *does not have* any extra work tape. Each tape has a read/write two way head. The work alphabet of all tapes is equal to the input alphabet $\mathbb{B}$. The input string (if $r = 1$) is delimited by two markers at the start and at the end of the string. The tape for this string has exactly that many cells that is needed to write the input string and the markers. The machine can read the markers but cannot write them. Also it cannot write anything to the cells containing the markers. We will consider two types of semi-infinite tapes (to write inputs from Ω):

- A semi-infinite tape with a marker at the beginning. Thus the head on the tape knows whether it is at the beginning. We call such tape *bounded*.
- A semi-infinite tape without any marker. The head on the tape does not know whether it is at the beginning of the tape. If this is the case and the head receives the command to move off the tape, the computation halts without any results (such moves are thus forbidden). We will call such tape *abrupt*.

As usually, the machine has a finite state unit controlling the heads. At the beginning of the computation all the heads are positioned at the first cells of the corresponding tapes (or in the cell number 0 for bi-infinite tapes) and the control unit is in the special starting state.

The main feature of ecological machines is that its work alphabet is equal to the input one. This prevents an ecological machine to simulate an extra work tape: such a simulation would corrupt the input. This explains also why we distinguish between two types of semi-infinite tapes. Indeed, ecological machines have no means to mark the beginning of the tape.

The power of ecological machines seems to be very restricted: to perform a computation, the machine should write something on its tapes, thus corrupting the input data. Therefore, it seems quite plausible that the ecological computational model is weaker than the usual one. That is, there is a (partial) function that is computable by a usual Turing machine of Definition 1 but is not computable by any ecological Turing machine of Definition 2. However, it turns out that for some r, s, t it is quite difficult to find out whether this is true or not. For example, we do not know whether ecological machines with exactly two bounded infinite tapes (thus $s \geq 2$, and r, t are arbitrary) are equivalent to usual ones. Moreover, in some cases the above hypothesis is false: we have proved that ecological machines with at least three bounded infinite tapes are as powerful as usual Turing machines (Theorem 6). On the other hand, we have proved that in the following cases ecological machines are weaker than usual ones:

- $s = 1$, $t = 0$, $r = 0, 1$ and the infinite tape is bounded (Theorem 4),
- s, t, r are arbitrary, but the machine does not have any bounded tapes (Theorem 5).

2 Total Functions

For total functions $f\colon \Omega \to \mathbb{B}$ the ecological model is equivalent to the usual one. Machines able to compute such functions have one semi-infinite tape, either abrupt or bounded. We will show that every total function of an infinite sequence actually depends only on a finite number of positions in this sequence.

Theorem 1. *Let $f\colon \Omega \to \mathbb{B}$ be a total function computable by a usual Turing machine (of Definition 1). Then f can be computed on an ecological Turing machine with one bounded or abrupt tape.*

Proof. Consider a usual Turing machine that computes f. Call a (finite) string x bad if that machine, in the run on input $x000\ldots$, will eventually move its head on the input tape beyond the string. Assume that there are infinitely many bad strings. Due to compactness of Ω, there is an infinite binary sequence having infinitely many bad prefixes. If we run the machine on this sequence, it will never halt, as it goes beyond every cell of the tape.

This contradiction shows that there is only a finite number of bad strings. So, the machine never moves its head on the input tape beyond some tape cell. Consequently, the function f depends only on a finite number of input cells. The contents of those cells form finitely many combinations. This allows to build an ecological machine computing f. We just plug into the program of the machine the table of these combinations and the value of f for each combination. □

Obviously, the theorem generalizes to total functions $f\colon \Omega^s \times \Psi^t \to \mathbb{B}$ for every s, t. Every such function depends only on a finite part of its input sequences, so it can be computed by an ecological Turing machine with abrupt or bounded tapes.

Consider now total functions $f\colon \mathbb{B}^* \times \Omega^s \times \Psi^t \to \mathbb{B}$. Such functions can be computed by ecological Turing machines with s semi-infinite tapes and t bi-infinite ones. Recall that we assume that $s + t > 0$. If all the semi-infinite tapes are abrupt then the ecological model is weaker than the usual one.

Theorem 2. *For every s, t (where $s + t > 0$) there exists a total function $f\colon \mathbb{B}^* \times \Omega^s \times \Psi^t \to \mathbb{B}$ that is computable in the usual sense, but is not computable by any ecological Turing machine whose semi-infinite tapes are abrupt.*

Proof. First, we prove the statement for $s = 1$, $t = 0$. Consider the following function $f(x, \alpha)$. Let k stand for the length of x. Let y denote the string consisting of the first k^2 characters of α. Let n stand for the natural number whose binary notation is written in $\lceil \log_2(k^2 + k) \rceil$ bits of α starting after position k^2. Then, define $f(x, \alpha)$ as the nth character of the string xy (where xy is the concatenation of x and y). This function is obviously computable in the usual model. We will prove that it is not computable in the ecological model.

Informally, ecological machines cannot compute f because while the head moves from the beginning of the tape to the place where n is written, the machine should store somewhere the distance between the current position of the head and the beginning of n. Thus it has to corrupt some part of xy. To formalize this idea we will use the notion of Kolmogorov complexity, which formalizes the notion of quantity of information.

The Kolmogorov complexity $K(w)$ of a string w is the length of the shortest program that prints w. The programming language (and the corresponding complexity K_0) can be chosen optimally. This means that for any other programming language (and corresponding complexity K_1) there exists a constant c such that $K_0(w) \le K_1(w) + c$ for all w. In the similar way, we define Kolmogorov complexity of natural numbers, pairs of strings, pairs of natural numbers, etc. By combinatorial arguments, in every finite set A there exists an element w such that $K(w) \ge \log_2 |A|$. Such elements of A are called *random elements of* A. For more details see [4].

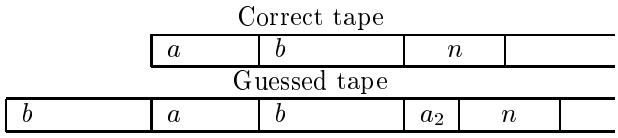
Assume that there is an ecological Turing machine computing f. Take sufficiently large k and a random triple $\langle x, y, P \rangle$, where x is a string of length k, y is a string of length k^2, and P is an integer in the range $1, \ldots, k^2$. Run the machine on the string x as its finite input, and on the sequence α consisting of y appended by zeroes, as its infinite input. Consider the first moment when the head on the infinite tape goes beyond the cell number P. If the machine never goes beyond P then it errs. Indeed, in this case the result of the machine does not depend on n; the string xy however has both 0s and 1s.

Let q be the internal state of the machine, l the position of the head on the finite tape, and w its content, at that moment. Let a stand for the content of the infinite tape from the beginning up to position P and b from the position P up to position k^2. Let ba denote the concatenation of b and a. We will prove that, given the tuple $T = \langle ba, q, l, w \rangle$, we can find xy and P. Since the triple $\langle x, y, P \rangle$ is random, its complexity is not less than $k + k^2 + 2 \log_2 k$. But the complexity of T is at most $k + k^2 + \log_2 k + \text{const}$. This cannot happen (for large enough k), so the theorem will be proven when we show that xy and P can actually be found.

We need to know the boundary between b and a in the string ba: If we know that boundary, we can simulate the computation of the machine starting from that moment for all $n = 1, \ldots, k^2 + k$ appended to y to find all bits of xy.

Guess some division of ba into b' and a'. Then simulate the run of the machine writing to the left of the head the string ba, to the right the string b', and then n, for all $n = 1, \ldots, k^2 + k$. As a result, we obtain a string $x'y'$ from the machine's answers.

Suppose that b' is longer than b, that is, $b' = ba_2$, $a = a_2a_1$ for some a_1, a_2. Then the correspondence between the correct tape and the guessed tape looks like this:



The head is between a and b on the correct tape, and between ba and b on the guessed one. Note that the machine have not yet seen any cells to the right of its current position. Also, it will never try to go off the left end of a, as the tape is abrupt. Consequently, its behavior is the same as its behavior on the tape aba_2n. Since the machine computes f, it regards the prefix of length $l = \lceil \log_2(k + k^2) \rceil$ of the string a_2n as (the binary notation of) a natural number n' and outputs n'th bit of the string xy. As a_2 is non-empty, the result of the machine does not depend of several (maybe even all) least significant bits of n. That is, the string $x'y'$ consists of blocks of 0's and of blocks of 1's (of length $\min\{2^{|a_2|}, k^2 + k\}$). This can be easily checked. If this is the case, the division is wrong. Indeed, we assumed that the triple $\langle x, y, P \rangle$ is random. Every

string xy consisting of blocks 00 and 11 can be described in $|xy|/2 + O(1)$ bits, and the whole triple $\langle x, y, P\rangle$ in $(k + k^2)/2 + l + O(1) \ll k + k^2 + l$ bits. Hence $\langle x, y, P\rangle$ is non-random in this case. The same holds when xy consists of longer blocks.

Thus we can decide whether b' is too long provided $|b'| \geq |b|$. Take the longest possible b', that is $b' = ba$, then one bit shorter and so on until we find out that $|b'| = |b|$.

Now we have found the correct division of ba into b and a and hence the value of P (as the length of a) and xy. Therefore, given the tuple $\langle ba, q, l, w\rangle$, which can be specified in $k + k^2 + \log_2 k$ bits, we can find the triple $\langle x, y, P\rangle$ of complexity at least $k + k^2 + 2\log_2 k$, a contradiction. Hence, there is no ecological machine with abrupt tape computing f.

Let us proceed to total functions $f\colon \mathbb{B}^* \times \Psi \to \mathbb{B}$. These functions can be computed on ecological machines with one finite tape and one bi-infinite tape. The counter example is similar to the previous case. More specifically, let k be the length of the input string x. Let y be the segment of α consisting of bits with numbers from $-k^2$ to k^2 (recall that α is bi-infinite). The number n will now be in the segment $[1, k + 2k^2 + 1]$, and again its binary notation will be written after y. The proof is modified as follows. The number P is again from 1 to k^2, but now we consider the first moment when the head moves out of the segment $[-P, P]$ on the infinite tape (in either direction). The rest of the proof is completely similar.

Now let us consider the general case. For the given s, t (where $s + t > 0$) we need to define a total computable function $f\colon \mathbb{B}^* \times \Omega^s \times \Psi^t \to \mathbb{B}$ that cannot be computed by any ecological machine having no bounded tapes. Let $f(x, \alpha_1, \ldots, \alpha_{s+t})$ be equal to nth bit of the word $z = xy_1 \ldots y_{s+t}$. Here, the strings $y_1, \ldots, y_{s+t}$ and the number n are defined as follows. Let $k = |x|$ and y_i the prefix of length k^2 of α_i, if $i \leq s$, and the segment of α_i consisting of characters with indices from $-k^2$ to k^2, if $i > s$. Let n_i be the number written on tape number i in positions from $k^2 + 1$ to $k^2 + \lceil \log_2 |z| \rceil$. The number n is equal to the sum of all n_i modulo $|z|$.

The above proof is modified as follows. We take a random tuple $\langle x, y_1, \ldots, y_{s+t}, P\rangle$, where P is a number in the range $1, \ldots, k^2$, and consider the first moment when some head on an infinite tape moves beyond position P (or $-P$ on a bi-infinite tape). The strings a_i and b_i are defined in the same way as above: a_i is the content of ith tape from the beginning to position of the head (from position $-k^2$ for bi-infinite tapes), and b_i is the content of the ith tape from the position of the head up to position k^2. We define q, l, w just as before.

To derive a contradiction we need to prove that z and P can be constructed given $\langle q, l, w, b_1 a_1, \ldots, b_{s+t} a_{s+t}\rangle$. To do this, we find a_i and b_i for all i in succession as follows. Try all possible divisions of $b_i a_i$ into a_i' and b_i' starting from the longest b_i'. For each division run the machine, writing on all tapes except ith one the strings $b_1 a_1, \ldots, b_{s+t} a_{s+t}$ both before the heads and after them. On ith tape write $b_i a_i$ before the head, and $b_i' n$ after it (for all $n \leq |z|$). If b_i' is longer than b_i, the resulting string z' will consist of blocks of 0's and 1's. Thus we will know that the division is incorrect, as, for the correct division, the string z' is a cyclic shift of z, so it cannot consist of such blocks. □

We can prove also that it is possible to fix the arguments $\alpha_1, \ldots, \alpha_{s+t}$ so that the function $f(x, \alpha_1, \ldots, \alpha_{s+t})$ defined in the proof of Theorem 2 is not computable (as the function of x) in the same ecological model.

Theorem 3. *There are sequences $\alpha_1, \ldots, \alpha_{s+t}$ such that the function $f(x, \alpha_1, \ldots, \alpha_{s+t})$ is not computable by any ecological Turing machine with abrupt tapes, whose tapes initially contain $\alpha_1, \ldots, \alpha_{s+t}$.*

Proof. We will prove the theorem in the case $s = 1$, $t = 0$ (one semi-infinite sequence). In other cases the proof is completely similar.

We will use the diagonalization argument. Fix an enumeration of all the ecological machines: $M_1, M_2, \ldots$.

Lemma 1. *There exist x, z such that for any α that has z as its prefix either the machine M_1 does not halt on input $\langle x, \alpha\rangle$, or it halts with a wrong answer.*

Proof. Choose k, x, y, P as in the proof of the previous theorem, and consider two cases.

First case: there is n such that for any sequence β the machine, in the run on input $x, yn\beta$, does not halt or tries to move the head off the tape. In this case let $z = yn$.

Second case: for every n there is β such that the machine, in the run on input $x, yn\beta$, does not try to move the head off the tape and halts. We claim that in this case there are n and β such that the machine, in

the run on input $x, yn\beta$, either tries to move the head off the tape or halts with a wrong result. Obviously, in this case the machine reads only a finite part v of the sequence β, thus we can let $z = ynv$.

Proof of the claim. Assume the contrary: for all n, β, the machine does not try to move the head off the tape and if it halts then it outputs the correct result. Then we can show that given $\langle ba, q, l, w\rangle$ we can construct xy as follows. Try all divisions of ba into b' and a', starting with empty a'. For each division try all n, and for each n look for β such that if we write on the tape a' (not ba as before!), then $b'n\beta$, and place the head between a' and b' then the machine will halt (either trying to move its head off the tape or not). Call a division *valid* if for each n such β exists. Obviously, the correct division is valid and, moreover, the machine outputs the correct result for every β. The assumption implies that every division with $|a'| < |a|$ is valid too. Just as in the proof of Theorem 2 the divisions with $|a'| < |a|$ can be filtered out by the same checking procedure: either for some n the machine tries to move the head beyond the tape or the resulting string $x'y'$ consists of blocks of 0's or 1's. Thus, starting with empty a', we try to prove that the division is valid, and then that the division is not filtered out by the checking procedure. The first division that is valid and is not filtered out is correct. □

Obviously, the statement of the lemma is valid for every ecological machine, not only for M_1. Moreover, the string z, whose existence is stated, can be chosen as an extension of any string z_0. Indeed, we need only that the complexity of the triple $\langle x, y, P\rangle$ is more than $k + k^2 + \log_2 k + \text{const}$. If we are restricted to y's that continue some string z_0, the logarithm of the number of possible triples $\langle x, y, P\rangle$ will decrease only by $|z_0|$. Thus, the complexity of a random triple will be equal to $k + k^2 + 2\log_2 k - |z_0|$, that is again larger than $k + k^2 + \log_2 k + \text{const}$ when k is sufficiently large.

Now we proceed as follows: we find x_1, z_1 such that the first machine M_1 fails for them, then we find x_2, z_2 such that the second machine M_2 fails for them, and such that z_1 is the prefix of z_2, etc. Then, we let α be the sequence that has all z_i as its prefixes. □

Using a more complicated counterexample we can prove that ecological machines with only *one* bounded tape are weaker than the usual ones.

Theorem 4. *There is a total computable function $f : \mathbb{B}^* \times \Omega \to \mathbb{B}$ that is not computable by ecological Turing machines with bounded semi-infinite tape.*

Proof. The function f is defined in almost the same way as in the proof of Theorem 2. The only difference is that in place of k^2 we take some function $g(k)$ that grows sufficiently fast. We will see further how much is this "sufficiently".

Suppose that the function f defined in this way is computable by some ecological machine with a bounded semi-infinite tape. Fix a large k and strings x, y of lengths $k, g(k)$, respectively. Assume that the string xy contains both ones and zeroes. Run the machine on the input x, y. At some moment t the head on the semi-infinite tape reaches the cell number $g(k) + 1$. We claim that t is much larger than $g(k)$.

Let us regard our machine M as a one-tape machine M' having $\text{const} \cdot k \cdot 2^k$ states (a state of M' is a tuple consisting of M's state, the content of M's finite tape, and the position of the head on that tape).

Lemma 2. *If a Turing machine has Q states, halts on all initial contents of the tape, and on some initial content reaches mth cell for the first time at the moment t, then*

$$t \geq \frac{m(\log_2 m - 1)}{2} \log_2 Q$$

Proof. Fix an initial content I of the tape such that the head reaches mth cell for the first time at the moment t. Let $i < m$ be a boundary between two consecutive tape cells. Consider the sequence of states of the machine at all the moments when the head crosses the boundary i (we consider the run up to the moment t only). This sequence is called the *trace at boundary*[1] i. We will prove that the traces at different boundaries are different (as sequences). Consequently, the average trace should have length of the order $\log_2 m / \log_2 Q$, and since t is equal to the sum of lengths of all traces, t cannot be smaller than $m \log_2 m / \log_2 Q$.

[1] The notion of trace was used in [2, 3] to prove that one-tape Turing machines cannot recognize the symmetry of input x in $o(|x|^2)$ steps.

Assume, for the contrary, that the traces at boundaries i and $i+j$, where $j>0$, are equal to the same sequence $q_1,\dots,q_l$. Let u be the part of initial tape content from the start of the tape up to position i, and v the part between positions i and $i+j$. We claim that the machine will never halt if the initial content of the tape is $I'=uvvv\dots$ Let C denote the initial computation, and let $t_1,\dots,t_l$ be the moments when the head crosses the boundary i, and $s_1,\dots,s_l$ the moments when the head crosses the boundary $i+j$, in C. Let $C[a,b]$ denote the part of the computation C from moment a to moment b, that is, the sequence of head moves, internal states and written symbols. Then the computation C' on $I'=uvvv\dots$ may be glued from parts of the computation C as follows:

$$\begin{aligned}C' =& C[1,t_k]C[t_k,s_1]\\ & C[t_1,t_2]C[s_2,s_3]C[t_3,t_4]\dots C[t_{k-2},t_{k-1}]C[s_{k-1},s_k]C[t_k,s_1]\\ & C[t_1,t_2]C[s_2,s_3]C[t_3,t_4]\dots C[t_{k-2},t_{k-1}]C[s_{k-1},s_k]C[t_k,s_1]\\ & \dots,\end{aligned}$$

where k is the largest number $k\le l$ such that $t_k<s_1$.

Let us verify this. Up to the moment t_k the computation C' is identical to C: $C'[1,t_k]=C[1,t_k]$, since up to the moment t_k the head never crosses the boundary $i+j$. Moreover, $C'[t_k,s_1]$ is also equal to $C[t_k,s_1]$. Consequently, the content of the tape in the segment $[1,i+j]$ are equal in both computations (up to the moment s_1). From that moment on, computations C' and C can differ.

Further, in computation C', the head comes beyond the boundary $i+j$ for the first time when the machine is in the state q_1. Since the initial content I' of the tape between $i+j$ and $i+2j$ is equal to v, the further behavior of the machine is identical to that in computation C from the moment t_1 up to the moment t_2, when the head crosses the boundary $i+j$ again, but now in the direction from right to left. The head sees in the segment $[i,i+j]$ what was written there during the computation $C[t_k,s_1]$ and enters this zone in the same state q_2 that it had in C at the moment s_2 (here we use for the first time the equality of the traces). Hence the computation C' further will be equal to $C[s_2,s_3]$ and the head will cross the boundary $i+j$ when machine is in state q_3. In zone $[i+j,i+2j]$ it sees the same symbols that were written in zone $[i,i+j]$ in the computation $C[t_1,t_2]$ and thus C' continues as $C[t_3,t_4]$, and so on.

After the moment t_k in C' the head will be in zone $[i,i+2j]$, crossing the $i+j$ boundary exactly k times in states $q_1,q_2,\dots,q_k$. After that it will cross the boundary $i+2j$ for the first time when the machine is in the state s_1. Here is the computation C' up to this moment (between parts of the computation we showed the boundary which is crossed by the head at that moment, the state of the machine and the direction of head movement):

$$\begin{aligned}C' =& C[1,t_k]\overrightarrow{\left[\begin{smallmatrix}i\\ q_n\end{smallmatrix}\right]}C[t_k,s_1]\overrightarrow{\left[\begin{smallmatrix}i+j\\ q_1\end{smallmatrix}\right]}\\ & C[t_1,t_2]\overleftarrow{\left[\begin{smallmatrix}i+j\\ q_2\end{smallmatrix}\right]}C[s_2,s_3]\overrightarrow{\left[\begin{smallmatrix}i+j\\ q_3\end{smallmatrix}\right]}C[t_3,t_4]\overleftarrow{\left[\begin{smallmatrix}i+j\\ q_4\end{smallmatrix}\right]}\dots C[t_{k-2},t_{k-1}]\overleftarrow{\left[\begin{smallmatrix}i+j\\ q_{k-1}\end{smallmatrix}\right]}C[s_{k-1},s_k]\overrightarrow{\left[\begin{smallmatrix}i+j\\ q_k\end{smallmatrix}\right]}C[t_k,s_1]\overrightarrow{\left[\begin{smallmatrix}i+2j\\ q_1\end{smallmatrix}\right]}\dots\end{aligned}\tag{2}$$

Another way to show that this is the case is to split the right hand side of (2) into two sequences: $C[1,t_k]$, $C[t_k,s_1]$, $C[s_2,s_3]$, $\dots$, $C[s_{k-1},s_k]$ and $C[t_1,t_2]$, $C[t_3,t_4]$, $\dots$, $C[t_{k-2},t_{k-1}]$, $C[t_k,s_1]$. The terms of the first sequence are just all the parts of C that happen in zone $[1,i+j]$. The terms of second sequence are just all the parts of $C[1,s_1]$ that happen in zone $[i,i+j]$.

Consequently, when the head in C' crosses the boundary $i+2j$, the content of zone $[i+j,i+2j]$ will be equal to the content of zone $[i,i+j]$ at the moment s_1 in C. Therefore, the cycle then repeats:

$$\overrightarrow{\left[\begin{smallmatrix}i+2j\\ q_1\end{smallmatrix}\right]}C[t_1,t_2]\overleftarrow{\left[\begin{smallmatrix}i+2j\\ q_2\end{smallmatrix}\right]}C[s_2,s_3]\overrightarrow{\left[\begin{smallmatrix}i+2j\\ q_3\end{smallmatrix}\right]}C[t_3,t_4]\overleftarrow{\left[\begin{smallmatrix}i+2j\\ q_4\end{smallmatrix}\right]}\dots C[t_{k-2},t_{k-1}]\overleftarrow{\left[\begin{smallmatrix}i+2j\\ q_{k-1}\end{smallmatrix}\right]}C[s_{k-1},s_k]\overrightarrow{\left[\begin{smallmatrix}i+2j\\ q_k\end{smallmatrix}\right]}C[t_k,s_1]\overrightarrow{\left[\begin{smallmatrix}i+3j\\ q_1\end{smallmatrix}\right]},$$

ad infinitum.

The number of different traces of length less than l is less than than Q^l. Hence for at least half the boundaries the trace is longer than $\frac{\log_2 m-1}{\log_2 Q}$. □

We have proved that for all x, y except for y that consist of 0s only or of 1s only the machine M reaches boundary $g(k)$ not faster than in $\frac{g(k)(\log_2 g(k)-1)}{2(k+\log_2 k+\mathrm{const})}$ steps. All the configurations of the machine up

to this moment are different (a configuration includes the state, the content of the finite tape and of the first $g(k)$ cells of the infinite tape, and positions of heads of both tapes). The sets of configurations for different pairs (x, y) pairs are disjoint too, because if we start the machine in each configuration and try all possible n, we can find both x and y.

Hence, the total number of configurations for all x, y is not less than

$$\frac{(2^{k+g(k)} - 2)g(k)(\log_2 g(k) - 1)}{2(k + \log_2 k + \text{const})}.$$

On the other hand, the total number of all possible configurations is

$$\text{const} \cdot 2^{k+g(k)} kg(k).$$

We see that for $g(k) = 2^{k^3}$ we get a contradiction if k is sufficiently large. □

3 Partial functions

For partial functions we can prove that the ecological model is weaker that the usual one even for $r = 0$ (without the finite input), if either there are no bounded tapes, or there is exactly one bounded semi-infinite tape, and no other infinite tapes.

Theorem 5. *For all s, t there is a partial function $g : \Omega^s \times \Psi^t \to \mathbb{B}$ that is computable in the usual model, but is not ecologically computable by machines with abrupt tapes. Also, there exists a partial function $g : \Omega \to \mathbb{B}$ that is computable in the usual model, but is not computable by ecological machines with semi-infinite bounded tape.*

Proof. For a given string x let $\bar{x}$ denote the string $00\ldots01x$, where the number of leading zeroes is equal to the length of x. The function g is defined for all sequences such that $\alpha_1(i) = 1$ for at least one $i \geq 1$. For any such sequence there exist the unique x and i such that $\alpha_1(1)\alpha_1(2)\ldots\alpha_1(i) = \bar{x}$. Cut the segment $[1, i]$ from α_1, and denote the rest of α_1 by β. The function g is now defined as $g(\alpha_1, \ldots, \alpha_{s+t}) = f(x, \beta, \alpha_2, \ldots, \alpha_{s+t})$, where f is the function defined in the proof of Theorem 2, where one should take 2^{2k} instead of k^2 (the description of x takes now $2k$ cells instead of k cells, and the logarithmic gap is not sufficient). The proof then goes as before.

The second statement is proved in a similar way. □

4 When the ecological machines are as strong as usual ones

(The idea belongs to An. A. Muchnik.) In this section, we study ecological machines with at least three bounded tapes. In this model, any computable function can be computed.

Theorem 6. *Any partial computable function $f\colon (\mathbb{B}^*)^r \times \Omega^s \times \Psi^t \to \mathbb{B}$ for $s \geq 3$ is ecologically computable by a machine with three bounded, $s - 3$ abrupt semi-infinite infinite tapes, and t bi-infinite tapes.*

Proof. We will assume that $r = 0$. For $r = 1$ the proof is similar.

First we define another computational model—a version of counter machines. The machine has a finite state control unit and n *counters*, each containing a natural number, initially zero. Besides, the machine has s semi-infinite abrupt tapes, and t bi-infinite tapes, all of them read-only. The control unit can increment and decrement counters (decrementing a counter whose value is zero does not change its value), and ask whether the value of a counter is zero. It can also move the heads on tapes in both directions and read tape symbols. The input to such a machine is s semi-infinite and t bi-infinite sequences, initially written on the tapes. We will these machines n *counter machines*, or machines with n counters.

Lemma 3. *For some constant n every partial computable function $f\colon \Omega^s \times \Psi^t \to \mathbb{B}$ is computable by a machine with n counters.*

Proof. If we have a sufficient number of counters, we can add, subtract, multiply, and divide numbers in two given counters. For example, to add numbers in counters A and B we need to decrease B simultaneously increasing A until B becomes zero. In a similar way we can implement subtraction; multiplication and division are implemented via addition and subtraction. So, we can check primality, and find primes sequentially in increasing order.

It suffices to simulate by a counter machine every machine described in Definition 1. To this end we need to keep the content of the work tape in one of the counters. We can do this by encoding each tape symbol as a number, and storing in a counter the number $2^{t_1}3^{t_2}5^{t_3}\dots p_N^{t_N}$, where 2, 3, …, p_N are consecutive primes, and t_1, t_2, …, t_N are codes of symbols of the tape of simulated machines. Another counter keeps the position of the head of the work tape of simulated machine. □

Obviously, we can assume that the simulating counter machine has the following property: at any moment in the computation every head except one is in the beginning of the tape (in cell number 1). Indeed, we can keep in counters the current positions of heads and hold the heads at the beginnings of tapes; when the counter machine needs to read a symbol of the tape, the corresponding head moves to the position stored in the counter, and then returns to the beginning of the tape. We will call such counter machines *correct*, and call the head that is not at the tape start *active*. If all heads are at the start, we call active the head that was last not at the start.

Lemma 4. *Every computable function* $f\colon (\mathbb{B}^*)^r \times \Omega^s \times \Psi^t \to \mathbb{B}$ *is computable by a correct machine with 2 counters.*

Proof. By Lemma lem-C-counters, for some n there is a correct machine with n counters that computes f. Encode values of counters into one number $S = 2^{c_1}3^{c_2}\dots p_n^{c_n}$, where, as before, p_i is ith prime number and c_i is the value of ith counter. We store the number S in the first of the two counters. The second one will be used for intermediate computations.

Now we have to show how to simulate operations with counters. Incrementing and decrementing ith counter correspond to multiplying and dividing S by p_i. Multiplying is done like this: while the first counter is not zero, decrement it and then p_i times increment the other counter. Division is done in a similar way. □

Thus, given a correct machine M_1 with two counters that computes f, we have to construct an ecological machine M_2 computing f. The simulation goes as follows. We will assume that $s = 3$ and for $s > 3$ the simulation is entirely similar. Fix a one-to-one correspondence between tapes of M_1 and M_2. The machine M_2 will not write anything on its tapes, but will only move the heads. The position of the active head of M_1 will be equal to the position of the corresponding head of M_2. Positions of two other heads of M_2 will be equal to the values of counters c_1 and c_2 of M_1. During the simulation, M_2 remembers which tape corresponds to which counter. For every counter the machine knows whether its value is zero, as all tapes are bounded and thus the knows whether the head is at the start.

The simulation now is as follows:

1 When M_1 increments or decrements the value of a counter, M_2 moves the corresponding head.

2 When M_1 moves the active head, M_2 does the same.

3 The remaining case is when all the heads of M_1 are at the start, and M_1 moves a non-active head. Let that be the first head and let the position of the first head of M_2 store the value of the counter c_1. At that moment, either the second or the third head of M_2 is not used to store the value of c_2. Let that be the second head. Then, M_2 assigns the second tape to hold the value of c_2, and "copies" it from the first tape to the second one by moving the first head toward the start while simultaneously moving the second head from the start (here we again need that the tapes of M_2 are bounded). Then, M_2 does the same that M_1 has done, i.e. moves the first head.

□

5 Open questions

Many questions about the relative power of usual and ecological computational models remain open. We will list some of them.

1. What is the power of ecological machines with two bounded tapes?

2. What about ecological machines with at least two infinite tapes, of which exactly one is a semi-infinite bounded tape? Are they weaker than usual machines?

3. Consider the machines with a finite input ($r = 1$), whose working time is limited by a polynomial in the length of finite input. Is it true that such machines are equivalent in computational power to the usual machines with the same time restriction? Theorems 2 and 3 give a partial answer to this question. But Theorem 6 used simulation with exponential overhead.

4. What can be said of the power of ecological RAM (Random access machines)? The input data to such a machine is an infinite sequence of natural numbers stored in its registers.

References

[1] V.N. Agafonov. Complexity of algorithms and computations: A course for students of University of Novosibirsk, part 1. Publishing house of University of Novosibirsk, 1975. (Russian)

[2] Ya.M. Barzdin. Complexity of symmetry recognition on Turing machines, Problemy kibernetiki, v. 15 (1965), 245–248. (Russian)

[3] F.C. Hennie. One tape off-line Turing machine computations. Information and Control, 8:6 (1965) 553–578.

[4] M. Li, P. Vitanyi. Introduction to Kolmogorov complexity and its applications. Springer Verlag, 1997.