View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Explore Bristol Research

Clifford, R., & Jalsenius, M. T. (2011). Lower Bounds for Online Integer Multiplication and Convolution in the Cell-Probe Model. In ICALP. (pp. 593-604). Springer. 10.1007/978-3-642-22006-7_50

Link to published version (if available):
10.1007/978-3-642-22006-7_50

Link to publication record in Explore Bristol Research
PDF-document

University of Bristol - Explore Bristol Research

General rights

This document is made available in accordance with publisher policies. Please cite only the published version using the reference above. Full terms of use are available:
http://www.bristol.ac.uk/pure/about/ebr-terms.html

Take down policy

Explore Bristol Research is a digital archive and the intention is that deposited content should not be removed. However, if you believe that this version of the work breaches copyright law please contact open-access@bristol.ac.uk and include the following information in your message:

• Your contact details
• Bibliographic details for the item, including a URL
• An outline of the nature of the complaint

On receipt of your message the Open Access Team will immediately investigate your claim, make an initial judgement of the validity of the claim and, where appropriate, withdraw the item in question from public view.

arXiv:1101.0768v4 [cs.DS] 23 Feb 2012

Tight Cell-Probe Bounds for Online Integer Multiplication and Convolution*

Raphaël Clifford† Markus Jalsenius†

Abstract

We show tight bounds for both online integer multiplication and convolution in the cell-probe model with word size w. For the multiplication problem, one pair of digits, each from one of two n digit numbers that are to be multiplied, is given as input at step i. The online algorithm outputs a single new digit from the product of the numbers before step $i+1$. We give a $\Theta\big(\frac{\delta}{w}\log n\big)$ bound on average per output digit for this problem where 2^δ is the maximum value of a digit. In the convolution problem, we are given a fixed vector V of length n and we consider a stream in which numbers arrive one at a time. We output the inner product of V and the vector that consists of the last n numbers of the stream. We show a $\Theta\big(\frac{\delta}{w}\log n\big)$ bound for the number of probes required per new number in the stream. All the bounds presented hold under randomisation and amortisation. Multiplication and convolution are central problems in the study of algorithms which also have the widest range of practical applications.

1 Introduction

We consider two related and fundamental problems: multiplying two integers and computing the convolution or cross-correlation of two vectors. We study both these problems in an online or streaming context and provide matching upper and lower bounds in the cell-probe model. The importance of these problems is hard to overstate with both the integer multiplication and convolution problems playing a central role in modern algorithms design and theory.

For notational brevity, we write $[q]$ to denote the set $\{0,\ldots,q-1\}$, where q is a positive integer.

Problem 1 (Online convolution)**.** *For a fixed vector $V \in [q]^n$ of length n, we consider a stream in which numbers from $[q]$ arrive one at a time. For each arriving number, before the next number arrives, we output the inner product (modulo q) of V and the vector that consists of the last n numbers of the stream.*

* A preliminary version of this paper appeared in ICALP '11.

† University of Bristol, Department of Computer Science, Bristol, U.K.

We show that there are instances of this problem such that any algorithm solving it will require $\Omega(\frac{\delta}{w} \log n)$ amortised time on average per output, where $\delta = \log_2 q$ and w is the number of bits per cell in the cell-probe model. The result is formally stated in Theorem 3.

Problem 2 (Online multiplication)**.** *Given two numbers $X, Y \in [q^n]$, where q is the base and n is the number of digits per number, we want to output the n least significant digits of the product of X and Y, in base q. We must do this under the constraint that the ith digit of the product (starting from the lower-order end) is outputted before the $(i+1)$th digit, and when the ith digit is outputted, we only have access to the i least significant digits of X and Y, respectively. We can think of the digits of X and Y arriving online in pairs, one digit from each of X and Y.*

We show that there are instances of this problem such that any algorithm solving it takes $\Omega(\frac{\delta}{w} \log n)$ time on average per input pair, where $\delta = \log_2 q$ and w is the number of bits per cell in the cell-probe model. The result is formally stated in Theorem 12

Our main technical innovation is to extend recently developed methods designed to give lower bounds on dynamic data structures to the seemingly distinct field of online algorithms. Where $\delta = w$, for example, we have $\Omega(\log n)$ lower bounds for both online multiplication and convolution, thereby matching the currently best known offline upper bounds in the RAM model. As we discuss in the Section 1.1, this may be the highest lower bound that can be formally proved for these problems without a further significant theoretical breakthrough.

For the convolution problem, one consequence of our results is a new separation between the time complexity of exact and inexact string matching in a stream. The convolution has played a particularly important role in the field of combinatorial pattern matching where many of the fastest algorithms rely crucially for their speed on the use of fast Fourier transforms (FFTs) to perform repeated convolutions. These methods have also been extended to allow searching for patterns in rapidly processed data streams [CEPP11, CS11]. The results we present here therefore give the first strict separation between the constant time complexity of online exact matching [Gal81] and any convolution based online pattern matching algorithm.

Although we show only the existence of probability distributions on the inputs for which we can prove lower bounds on the expected running time of any deterministic algorithm, by Yao's minimax principle [Yao77] this also immediately implies that for every (randomised) algorithm, there is a worst-case input such that the (expected) running time is equally high. Therefore our lower bounds hold equally for randomised algorithms as for deterministic ones.

The lower bounds we show for both online multiplication and convolution are also tight within the cell-probe model. This can be seen by application of reductions described in [FS73, CEPP11]. It was shown there that any offline algorithm for multiplication [FS73] or convolution [CEPP11] can be converted to an online one with at most an $O(\log n)$ factor overhead. For details of these reductions we refer the reader to the original papers. In our case, the same approach also allows us to directly convert any

cell-probe algorithm from an offline to online setting. An offline cell-probe algorithm for either multiplication or convolution could first read the whole input, then compute the answers and finally output them. This takes $O(\frac{\delta}{w}n)$ cell probes. We can therefore derive online cell-probe algorithms which take only $O(\frac{\delta}{w}\log n)$ probes per output, hence matching the new lower bound we give.

1.1 Previous results and upper bounds in the RAM model

The best time complexity lower bounds for online multiplication of two n-bit numbers were given in the 1974 by Paterson, Fischer and Meyer. They presented an $\Omega(\log n)$ lower bound for multitape Turing machines [PFM74] and also gave an $\Omega(\log n/\log\log n)$ lower bound for the 'bounded activity machine' (BAM). The BAM, which is a strict generalisation of the Turing machine model but which has nonetheless largely fallen out of favour, attempts to capture the idea that future states can only depend on a limited part of the current configuration. To the authors' knowledge, there has been no progress on cell-probe lower bounds for online multiplication or convolution previous to the work we present here.

There have however been attempts to provide offline lower bounds for the related problem of computing the FFT. In [Mor73] Morgenstern gave an $\Omega(n\log n)$ lower bound conditional on the assumption that the underlying field of the transform is the complex numbers and that the modulus of any complex numbers involved in the computation is at most 1. Papadimitriou gave the same $\Omega(n\log n)$ lower bound for FFTs of length a power of two, this time excluding certain classes of algorithms including those that rely on linear mathematical relations among the roots of unity [Pap79]. This work had the advantage of giving a conditional lower bound for FFTs over more general algebras than was previously possible, including for example finite fields. In 1986 Pan [Pan86] showed that another class of algorithms having a so-called synchronous structure must require $\Omega(n\log n)$ time for the computation of both the FFT and convolution.

The fastest known algorithms for both offline integer multiplication and convolution in the word-RAM model require $O(n\log n)$ time by a well known application of a constant number of FFTs. As a consequence our online lower bounds match the best known time upper bounds for the offline problem. As we discussed above, our lower bounds are also tight within the cell-probe model for the online problems. The question now naturally arises as to whether one can find higher lower bounds in the RAM model. This appears as an interesting question as there remains a gap between the best known time upper bounds provided by existing algorithms and the lower bounds that we give within the cell-probe model. However, as we mention above, any offline algorithm for convolution or multiplication can be converted to an online one with at most an $O(\log n)$ factor overhead [FS73, CEPP11]. As a consequence, it is likely to be hard to prove a higher lower bound for the online problem than we have given, at least for the case where $\delta/w \in \Theta(1)$, as this would immediately imply a superlinear lower bound for offline convolution or multiplication. Such superlinear lower bounds are not yet known for any problem in **NP** except in very restricted models of computation, such as for example a single tape Turing Machine. Our only alternative route to find tight time bounds

would be to find better upper bounds for the online problems. For the case of online multiplication at least, this has been an open problem since at least 1973 and has so far resisted our best attempts.

1.2 The cell-probe model

When stating lower bounds it is important to be precise about the model in which the bounds apply. Our bounds in this paper hold in perhaps the strongest model of them all, the *cell-probe model*, introduced originally by Minsky and Papert [MP69, MP88] in a different context and then subsequently by Fredman [Fre78] and Yao [Yao81]. In this model, there is a separation between the computing unit and the memory, which is external and consists of a set of cells of w bits each. The computing unit cannot remember any information between operations. Computation is free and the cost is measured only in the number of cell reads or writes (cell-probes). This general view makes the model very strong, subsuming for instance the popular word-RAM model. In the word-RAM model certain operations on words, such as addition, subtraction and possibly multiplication take constant time (see for example [Hag98] for a detailed introduction). Here a word corresponds to a cell. Typically we think of the cell size w as being at least $\log_2 n$ bits, where n is the number of cells. This allows each cell to hold the address of any location in memory.

The generality of the cell-probe model makes it particularly attractive for establishing lower bounds for data structure problems and many such results have been given in the past couple of decades. The approaches taken have until recently mainly been based on communication complexity arguments and the chronogram technique of Fredman and Saks [FS89]. There remains however, a number of unsatisfying gaps between the lower bounds and known upper bounds. Only a few years ago, a breakthrough lead by Demaine and Pătraşcu gave us the tools to seal the gaps for several data structure problems [PD06]. The new technique was based on information theoretic arguments. Demaine and Pătraşcu also presented ideas which allowed them to express more refined lower bounds such as trade-offs between updates and queries of dynamic data structures. For a list of data structure problems and their lower bounds using these and related techniques, see for example [Păt08].

1.3 Organisation

We present the new cell-probe lower bound for online convolution in Section 2 along with the main techniques that we will use throughout. In Section 3 we show how these can then be applied to the problem of online multiplication.

2 Online convolution

For a vector V of length n and $i \in [n]$, we write $V[i]$ to denote the elements of V. For positive integers n and q, the *inner product* of two vectors $U, V \in [q]^n$, denoted $\langle U, V \rangle$,

is defined as

$$\langle U, V \rangle = \sum_{i \in [n]} (U[i] \cdot V[i]) \,.$$

Parameterised by two positive integers n and q, and a fixed vector $V \in [q]^n$, the *online convolution problem* asks to maintain a vector $U \in [q]^n$ subject to an operation next(Δ), which takes a parameter $\Delta \in [q]$, modifies U to be the vector $(U[1], U[2], \ldots, U[n-1], \Delta)$ and then returns the inner product $\langle U, V \rangle$. In other words, next(Δ) modifies U by shifting all elements one step to the left, pushing the leftmost element out, and setting the new rightmost element to Δ. We consider the online convolution problem over the ring $\mathbb{Z}/q\mathbb{Z}$, that is integer arithmetic modulo q. Let $\delta = \log_2 q$.

Theorem 3. *For any positive integers q and n, in the cell probe model with w bits per cell there exist instances of the online convolution problem such that the expected amortised time per* next*-operation is $\Omega\left(\frac{\delta}{w} \log n\right)$, where $\delta = \log_2 q$.*

In order to prove Theorem 3 we will consider a random instance that is described by n next-operations on the sequence $\Delta = (\Delta_0, \ldots, \Delta_{n-1})$, where each Δ_i is chosen independently and uniformly at random from $[q]$. We defer the choice of the fixed vector V until later. For t from 0 to $n-1$, we use t to denote the time, and we say that the operation next(Δ_t) occurs at time t.

We may assume that prior to the first update, the vector $U = \{0\}^n$, although any values are possible since they do not influence the analysis. To avoid technicalities we will from now on assume that n is a power of two.

2.1 Information transfer

Following the overall approach of Demaine and Pătraşcu [PD04] we will consider adjacent time intervals and study the *information* that is transferred from the operations in one interval to the next interval. More precisely, let $t_0, t_1, t_2 \in [n]$ such that $t_0 \leqslant t_1 < t_2$ and consider any algorithm solving the online convolution problem. We would like to keep track of the memory cells that are written to during the time interval $[t_0, t_1]$ and then read during the succeeding interval $[t_1+1, t_2]$. The information from the next-operations taking place in the interval $[t_0, t_1]$ that the algorithm passes on to the interval $[t_1+1, t_2]$ must be contained in these cells. Informally one can say that there is no other way for the algorithm to determine what occurred during the interval $[t_0, t_1]$ except through these cells. Formally, the *information transfer*, denoted $IT(t_0, t_1, t_2)$, is defined to be the set of memory cells c such that c is written during $[t_0, t_1]$, read at a time $t_r \in [t_1+1, t_2]$ and not written during $[t_1+1, t_r]$. Hence a cell that is overwritten in $[t_1+1, t_2]$ before being read is not included in the information transfer. Observe that the information transfer depends on the algorithm, the vector V and the sequence Δ. The first aim is to show that for any choice of algorithm solving the online convolution problem, the number of cells in the information transfer is bounded from below by a sufficiently large number for some choice of the vector V.

For $0 \leqslant t_0 \leqslant t_1 < n$, we write $\Delta_{[t_0,t_1]}$ to denote the subsequence $(\Delta_{t_0}, \ldots, \Delta_{t_1})$ of Δ, and $\Delta_{[t_0,t_1]^c}$ to denote the sequence $(\Delta_0, \ldots, \Delta_{t_0-1}, \Delta_{t_1+1}, \ldots, \Delta_{n-1})$ which contains

all the elements of Δ except for those in $\Delta_{[t_0,t_1]}$. For $t \in [n]$, we let $P_t \in [q]$ denote the inner product returned by next(Δ_t) at time t (recall that we operate modulo q). We let $P_{[t_1+1,t_2]} = (P_{t_1+1}, \ldots, P_{t_2})$.

Since Δ is a random variable, so is $P_{[t_1+1,t_2]}$. In particular, if we condition on a fixed choice of $\Delta_{[t_0,t_1]^c}$, call it $\Delta^{\text{fix}}_{[t_0,t_1]^c}$, then $P_{[t_1+1,t_2]}$ is a random variable that depends on the random values in $\Delta_{[t_0,t_1]}$. The dependency on the next-operations in the interval $[t_0, t_1]$ is captured by the information transfer $IT(t_0, t_1, t_2)$, which must encode all the relevant information in order for the algorithm to correctly output the inner products in $[t_1 + 1, t_2]$. In other words, an encoding of the information supplied by cells in the information transfer is an upper bound on the conditional *entropy* of $P_{[t_1+1,t_2]}$. This fact is stated in Lemma 4 and was given in [Păt08] with small notational differences.

Lemma 4 (Lemma 3.2 of [Păt08])**.** *The entropy*

$$H(P_{[t_1+1,t_2]} \mid \Delta_{[t_0,t_1]^c} = \Delta^{\text{fix}}_{[t_0,t_1]^c}) \leqslant \\ w + 2w \cdot \mathbb{E}\left[|IT(t_0, t_1, t_2)| \mid \Delta_{[t_0,t_1]^c} = \Delta^{\text{fix}}_{[t_0,t_1]^c}\right].$$

Proof. The average length of any encoding of $P_{[t_1+1,t_2]}$ (conditioned on $\Delta^{\text{fix}}_{[t_0,t_1]^c}$) is an upper bound on its entropy. We use the information transfer as an encoding in the following way. For every cell c in the information transfer $IT(t_0, t_1, t_2)$, we store the address of c, which takes at most w bits under the assumption that the cell size can hold the address of every cell, and we store the contents of c, which is a cell of w bits. In total this requires $2w \cdot |IT(t_0, t_1, t_2)|$ bits. In addition, we store the size of the information transfer, $|IT(t_0, t_1, t_2)|$, so that any algorithm decoding the stored information knows how many cells are stored and hence when to stop checking for stored cells. Storing the size of the information transfer requires w bits, thus the average total length of the encoding is $w + 2w \cdot \mathbb{E}[|IT(t_0, t_1, t_2)| \mid \Delta_{[t_0,t_1]^c} = \Delta^{\text{fix}}_{[t_0,t_1]^c}]$.

In order to prove that the described encoding is valid, we describe how to decode the stored information. We do this by simulating the algorithm. First we simulate the algorithm from time 0 to $t_0 - 1$. We have no problem doing so since all necessary information is available in $\Delta^{\text{fix}}_{[t_0,t_1]^c}$, which we know. We then skip from time t_0 to t_1 and resume simulating the algorithm from time $t_1 + 1$ to t_2. In this interval, the algorithm outputs the values in $P_{[t_1+1,t_2]}$. In order to correctly do so, the algorithm might need information from the next-operations in $[t_0, t_1]$. This information is only available through the encoding described above. When simulating the algorithm, for each cell c we read, we check if the address of c is contained in the list of addresses that was stored. If so, we obtain the contents of c by reading its stored value. Each time we write to a cell whose address is in the list of stored addresses, we remove it from the stored list, or blank it out. Note that every cell we read whose address is not in the stored list contains a value that was written last either before time t_0 or after time t_1. Hence its value is known to us. □

2.2 Recovering information

In the previous section, we provided an upper bound for the entropy of the outputs from the next-operations in $[t_1+1, t_2]$. Next we will explore how much information *needs to be* communicated from $[t_0, t_1]$ to $[t_1+1, t_2]$. This will provide a lower bound on the entropy. As we will see, the lower bound can be expressed as a function of the length of the intervals and the vector V.

Suppose that $[t_0, t_1]$ and $[t_1+1, t_2]$ both have the same length ℓ. That is, $t_1 - t_0 + 1 = t_2 - t_1 = \ell$. For $i \in [\ell]$, the output at time $t_1 + 1 + i$ can be broken into two sums S_i and S'_i, such that $P_{t_1+1+i} = S_i + S'_i$, where

$$S_i = \sum_{j \in [\ell]} (V[n-1-(\ell+i)+j] \cdot \Delta_{t_0+j})$$

is the contribution from the alignment of V with $\Delta_{[t_0,t_1]}$, and S'_i is the contribution from the alignments that do not include $\Delta_{[t_0,t_1]}$.

We define $M_{V,\ell}$ to be the $\ell \times \ell$ matrix with entries $M_{V,\ell}(i,j) = V[n-1-(\ell+i)+j]$. That is,

$$M_{V,\ell} = \begin{pmatrix} V[n-\ell-1] & V[n-\ell+0] & V[n-\ell+1] & \cdots & V[n-2] \\ V[n-\ell-2] & V[n-\ell-1] & V[n-\ell+0] & \cdots & V[n-3] \\ V[n-\ell-3] & V[n-\ell-2] & V[n-\ell-1] & \cdots & V[n-4] \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ V[n-2\ell] & V[n-2\ell+1] & V[n-2\ell+2] & \cdots & V[n-\ell-1] \end{pmatrix}.$$

We observe that $M_{V,\ell}$ is a *Toeplitz* matrix (or "upside down" *Hankel* matrix) since it is constant on each descending diagonal from left to right. This property will be important later. From the definitions above it follows that

$$M_{V,\ell} \times \begin{pmatrix} \Delta_{t_0} \\ \Delta_{t_0+1} \\ \vdots \\ \Delta_{t_1} \end{pmatrix} = \begin{pmatrix} S_0 \\ S_1 \\ \vdots \\ S_{\ell-1} \end{pmatrix}. \tag{1}$$

We define the *recovery number* $R_{V,\ell}$ to be the number of variables $x \in \{x_1, \ldots, x_\ell\}$ such that x can be determined uniquely by the system of linear equations

$$M_{V,\ell} \times \begin{pmatrix} x_1 \\ \vdots \\ x_\ell \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_\ell \end{pmatrix},$$

where we operate in $\mathbb{Z}/q\mathbb{Z}$. The recovery number may be distinct from the rank of a matrix, even where we operate over a field. As an example, consider the all ones matrix. The matrix will have recovery number zero but rank one. The recovery number is however related to the conditional entropy of $P_{[t_1+1,t_2]}$ as described by the next lemma.

Lemma 5. *If the intervals $[t_0, t_1]$ and $[t_1 + 1, t_2]$ both have the same length ℓ, then the entropy*

$$H(P_{[t_1+1,t_2]} \mid \Delta_{[t_0,t_1]^c} = \Delta^{\text{fix}}_{[t_0,t_1]^c}) \geqslant \delta R_{V,\ell}\,.$$

Proof. As described above, for $i \in [\ell]$, $P_{t_1+1+i} = S_i + S'_i$, where S'_i is a constant that only depends on V and $\Delta^{\text{fix}}_{[t_0,t_1]^c}$. Hence we can compute the values $S_0, \ldots, S_{\ell-1}$ from $P_{[t_1+1,t_2]}$. From Equation (1) it follows that $S_0, \ldots, S_{\ell-1}$ uniquely specify $R_{V,\ell}$ of the parameters in $\Delta_{[t_0,t_1]}$. That is, we can recover $R_{V,\ell}$ of the parameters from the interval $[t_0, t_1]$. Each of these parameters is a random variable that is uniformly distributed in $[q]$, so it contributes δ bits of entropy. □

We now combine Lemmas 4 and 5 in the following corollary.

Corollary 6. *For any fixed vector V, two intervals $[t_0, t_1]$ and $[t_1 + 1, t_2]$ of the same length ℓ, and any algorithm solving the online convolution problem on Δ chosen uniformly at random from $[q]^n$,*

$$\mathbb{E}\left[|IT(t_0, t_1, t_2)|\right] \geqslant \frac{\delta R_{V,\ell}}{2w} - \frac{1}{2}\,.$$

Proof. For $\Delta_{[t_0,t_1]^c}$ fixed to $\Delta^{\text{fix}}_{[t_0,t_1]^c}$, comparing Lemmas 4 and 5, we see that

$$\mathbb{E}\left[|IT(t_0, t_1, t_2)| \mid \Delta_{[t_0,t_1]^c} = \Delta^{\text{fix}}_{[t_0,t_1]^c}\right] \geqslant \frac{\delta R_{V,\ell}}{2w} - \frac{1}{2}\,.$$

The result follows by taking expectation over $\Delta_{[t_0,t_1]^c}$ under the random sequence Δ. □

2.3 The lower bound for online convolution

We now show how a lower bound on the total number of cell reads over n next-operations can be obtained by summing the information transfer between many pairs of time intervals. We again follow the approach of Demaine and Pătraşcu [PD04], which involves conceptually constructing a balanced tree over the time axis. This *lower bound tree*, denoted $\mathcal{T}$, is a balanced binary tree on n leaves. Recall that we have assumed that n is a power of two. The leaves, from left to right, represent the time t from 0 to $n - 1$, respectively. An internal node v is associated with the times t_0, t_1 and t_2 such that the two intervals $[t_0, t_1]$ and $[t_1 + 1, t_2]$ span the left subtree and the right subtree of v, respectively. For example, in Figure 1, the node labelled v is associated with the intervals $[16, 23]$ and $[24, 31]$.

For an internal node v of $\mathcal{T}$, we write $IT(v)$ to denote $IT(t_0, t_1, t_2)$, where t_0, t_1, t_2 are associated with v. We write $L(v)$ to denote the number of leaves in the left (same as the right) subtree of v. The key lemma, stated next, is a modified version of Theorem 3.6 in [Păt08]. The statement of the lemma is adapted to our online convolution problem and the proof relies on Corollary 6.

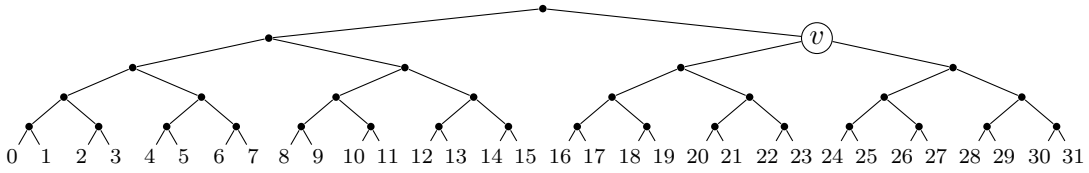

Figure 1: A lower bound tree $\mathcal{T}$ over $n = 32$ operations.

Lemma 7. *For any fixed vector V and any algorithm solving the online convolution problem, the expected running time of the algorithm over a sequence Δ that is chosen uniformly at random from $[q]^n$ is at least*

$$\frac{\delta}{2w} \sum_{v \in \mathcal{T}} R_{V,L(v)} - \frac{n-1}{2},$$

where the sum is over the internal nodes of $\mathcal{T}$.

Proof. We first consider a fixed sequence Δ. We argue that the number of read instructions executed by the algorithm is at least $\sum_{v \in \mathcal{T}} |IT(v)|$. To see this, for any read instruction, let t_r be the time it is executed. Let $t_w \leqslant t_r$ be the time the cell was last written, ignoring $t_r = t_w$. Then this read instruction (the cell it acts upon), is contained in $IT(v)$, where v is the lowest common ancestor of t_w and t_r. Thus, $\sum_{v \in \mathcal{T}} |IT(v)|$ never double-counts a read instruction.

For a random Δ, an expected lower bound on the number of read instructions is therefore $\mathbb{E}[\sum_{v \in \mathcal{T}} |IT(v)|]$. Using linearity of expectation and Corollary 6, we obtain the lower bound in the statement of the lemma. □

2.3.1 Lower bound with a random vector V

We have seen in Lemma 7 that a lower bound is highly dependent on the recovery numbers of the vector V. In the next lemma, we show that a random vector V has recovery numbers that are large.

Lemma 8. *Suppose that q is a prime and the vector V is chosen uniformly at random from $[q]^n$. Then $\mathbb{E}[R_{V,\ell}] \geqslant \ell/2$ for every length ℓ.*

Proof. Recall that $M_{V,\ell}$ is an $\ell \times \ell$ Toeplitz matrix. It has been shown in [KL96] that for any ℓ, out of all the $\ell \times \ell$ Toeplitz matrices over a finite field of q elements, a fraction of exactly $(1 - 1/q)$ is non-singular. This fact was actually already established in [Day60] almost 40 years earlier but incidentally reproved in [KL96]. Since we have assumed in the statement of the lemma that q is a prime, the ring $\mathbb{Z}/q\mathbb{Z}$ we operate in is indeed a finite field. The diagonals of $M_{V,\ell}$ are independent and uniformly distributed in $[q]$, hence the probability that $M_{V,\ell}$ is invertible is $(1 - 1/q) \geqslant 1/2$. If $M_{V,\ell}$ is invertible then the recovery number $R_{V,\ell} = \ell$; there is a unique solution to the system of linear equations in Equation (1). On the other hand, if $M_{V,\ell}$ is not invertible then the recovery

number will be lower. Thus, we can safely say that the expected recovery number $R_{V,\ell}$ is at least $\ell/2$, which proves the lemma. □

Before we give a lower bound for a random choice of V in Theorem 10 below, we state the following fact.

Fact 9. *For a balanced binary tree with n leaves, the sum of the number of leaves in the subtree rooted at v, taken over all internal nodes v, is $n \log_2 n$.*

Theorem 10. *Suppose that q is a prime. In the cell-probe model with w bits per cell, any algorithm solving the online convolution problem on a vector V and Δ, both chosen uniformly at random from $[q]^n$, will run in $\Omega\left(\frac{\delta}{w} n \log n\right)$ time in expectation, where $\delta = \log_2 q$.*

Proof. For a random vector V, a lower bound is obtained by taking the expectation of the bound in the statement of Lemma 7. Using linearity of expectation and applying Lemma 8 and Fact 9 completes the proof. □

Remark. Theorem 10 requires that q is a prime but for an integer $\delta > 1$, $q = 2^\delta$ is not a prime. However, we know that there is always at least one prime p such that $2^{\delta-1} < p < 2^\delta$. Thus, Theorem 10 is applicable for any integer δ, only with an adjustment by at most one.

2.3.2 Lower bound with a fixed vector V

We demonstrate next that it is possible to design a fixed vector V with guaranteed large recovery numbers. We will use this vector in the proof of Theorem 3. The idea is to let V consist of stretches of 0s interspersed by 1s. The distance between two succeeding 1s is an increasing power of two, ensuring that for half of the alignments in the interval $[t_1 + 1, t_2]$, all but exactly one element of $\Delta_{[t_0,t_1]}$ are simultaneously aligned with a 0 in V. We define the binary vector $K_n \in [2]^n$ to be

$$K_n = (\ldots 000000000000\mathbf{1}0000000000000000\mathbf{1}00000001000\mathbf{1}0\mathbf{11}0),$$

or formally,

$$K_n[i] = \begin{cases} 1, & \text{if } n-1-i \text{ is a power of two;} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Lemma 11. *Suppose $V = K_n$ and $\ell \geqslant 1$ is a power of two. The recovery number $R_{V,\ell} \geqslant \ell/2$.*

Proof. Recall that entry $M_{V,\ell}(i,j) = V[n-1-(\ell+i)+j]$. Thus, $M_{V,\ell}(i,j) = 1$ if and only if $n-1-(n-1-(\ell+i)+j) = \ell+i-j$ is a power of two. It follows that for row $i = \ell/2, \ldots, \ell-1$, $M_{V,\ell}(i,j) = 1$ for $j = i$ and $M_{V,\ell}(i,j) = 0$ for $j \neq i$. This implies that the recovery number $R_{V,\ell}$ is at least $\ell/2$. □

We finally give the proof of Theorem 3.

Theorem 3. We assume that n is a power of two. Let $V = K_n$. It follows from Lemma 11 and Fact 9 that $\sum_{v\in\mathcal{T}} R_{V,L(v)} \geqslant \sum_{v\in\mathcal{T}} L(v)/2 = \Omega(n \log n)$. Note that $L(v)$ is a power of two for every node v in $\mathcal{T}$. For Δ chosen uniformly at random from $[q]^n$, apply Lemma 7 to obtain the expected running time $\Omega\left(\frac{\delta}{w} n \log n\right)$ over n next-operations. □

3 Online multiplication

In this section we consider online multiplication of two n-digit numbers in base $q \geqslant 2$. For a non-negative integer X, let $X[i]$ denote the ith digit of X in base q, where the positions are numbered starting with 0 at the right (lower-order) end. We think of X padded with zeros to make sure that $X[i]$ is defined for arbitrarily large i. For $j \geqslant i$, we write $X[i\,..\,j]$ to denote the integer that is written $X[j]\cdots X[i]$ in base q. For example, let $X = 15949$ (decimal representation) and $q = 8$ (octal):

$$\begin{aligned} &X = 37115 \text{ (base 8)} && X[0] = 5 \\ &X[1\,..\,3] = 711 \text{ (base 8)} = 457 \text{ (decimal)} && X[3] = 7 \\ &X[3\,..\,10] = 37 \text{ (base 8)} = 31 \text{ (decimal)} && X[15] = 0 \end{aligned}$$

The *online multiplication problem* is defined as follows. The input is two n-digit numbers $X, Y \in [q^n]$ in base q (higher order digits may be zero). Let $Z = X \times Y$. We want to output the n lower order digits of Z in base q (i.e. $Z[0], \ldots, Z[n-1]$) under the constraint that $Z[i]$ must be outputted before $Z[i+1]$ and when $Z[i]$ is outputted, we are not allowed to use any knowledge of the digits $X[i+1], \ldots, X[n-1]$ and $Y[i+1], \ldots, Y[n-1]$. We can think of the digits of X and Y arriving one pair at a time, starting with the least significant pair of digits, and we output the corresponding digit of the product of the two numbers seen so far.

We also consider a variant of the online multiplication problem when one of the two input numbers, say Y, is known in advance. That is, all its digits are available at every stage of the algorithm and only the digits of X arrive in an online fashion. In particular we will consider the case when $Y = K_{q,n}$ is fixed, where we define $K_{q,n}$ to be the largest number in $[q^n]$ such that the ith bit in the binary expansion of $K_{q,n}$ is 1 if and only if i is a power of two (starting with $i = 0$ at the lower-order end). We can see that the binary expansion of $K_{q,n}$ is the reverse of $K_{(n \log_2 q)}$ in Equation (2). We will prove the following result.

Theorem 12. *For any positive integers δ and n in the cell probe model with w bits per cell, the expected running time of any algorithm solving the online multiplication problem on two n-digit random numbers $X, Y \in [q^n]$ is $\Omega(\frac{\delta}{w} n \log n)$, where $q = 2^\delta$ is the base. The same bound holds even under full access to every digit of Y, and when $Y = K_{q,n}$ is fixed.*

It suffices to prove the lower bound for the case when we have full access to every digit of Y; we could always ignore digits. We prove Theorem 12 using the same approach as for the online convolution problem. Here the next-operation delivers a new digit of

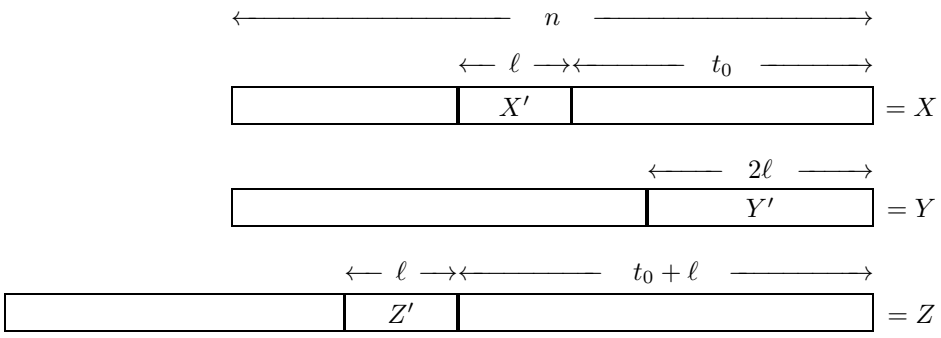

Figure 2: X, Y and $Z = X \times Y$ in base q.

X, which is chosen uniformly at random from $[q]$, and outputs the corresponding digit of the product of X and Y.

For $t_0, t_1, t_2 \in [n]$ such that $t_0 \leqslant t_1 < t_2$, we write $X[t_0, t_1]^{\mathrm{c}}$ to denote every digit of X (in base q) except for those at position t_0 through t_1. It is helpful to think of $X[t_0, t_1]^{\mathrm{c}}$ as a vector of digits rather than a single number. We write $X^{\mathrm{fix}}[t_0, t_1]^{\mathrm{c}}$ to denote a fixed choice of $X[t_0, t_1]^{\mathrm{c}}$. During the interval $[t_1 + 1, t_2]$, we output $Z[(t_1 + 1) \,..\, t_2]$. The information transfer is defined as before, and Lemma 4 is replaced with the following lemma.

Lemma 13. *The entropy*

$$H(Z[(t_1+1)\,..\,t_2] \mid X[t_0,t_1]^{\mathrm{c}} = X^{\mathrm{fix}}[t_0,t_1]^{\mathrm{c}}) \leqslant w + 2w \cdot \mathbb{E}\left[|IT(t_0,t_1,t_2)| \mid X[t_0,t_1]^{\mathrm{c}} = X^{\mathrm{fix}}[t_0,t_1]^{\mathrm{c}}\right].$$

3.1 Retrorse numbers and the lower bound

In Figure 2, the three numbers X, Y and $Z = X \times Y$ are illustrated with some segments of their digits labelled X', Y' and Z'. Informally, we say that Y' is *retrorse* if Z' depends heavily on X'. We have borrowed the term from Paterson, Fischer and Meyer [PFM74], however, we give it a more precise meaning, formalised below.

Suppose $[t_0, t_1]$ and $[t_1+1, t_2]$ both have the same length ℓ. For notational brevity, we write X' to denote $X[t_0 \,..\, t_1]$, Y' to denote $Y[0 \,..\, (2\ell-1)]$ and Z' to denote $Z[(t_1+1) \,..\, t_2]$ (see Figure 2). We say that Y' is *retrorse* if for any fixed values of t_0, $X[t_0, t_1]^{\mathrm{c}}$ (the digits of X outside $[t_0, t_1]$) and $Y[2\ell \,..\, (n-1)]$, each value of Z' can arise from at most four different values of X'. That is to say there is at most a four-to-one mapping from possible values of X' to possible values of Z'. We define $I_{Y,\ell} = \ell$ if Y' is retrorse, otherwise $I_{Y,\ell} = 0$. Note that $I_{Y,\ell}$ only depends on Y and ℓ. We will use $I_{Y,\ell}$ similarly to the recovery number $R_{V,\ell}$ from Section 2.2 and replace Lemma 5 with Lemma 14 below, which combined with Lemma 13 gives us Corollary 15.

Lemma 14. *If the intervals $[t_0, t_1]$ and $[t_1 + 1, t_2]$ both have the same length ℓ, then the entropy*

$$H(Z[(t_1+1)\,..\,t_2] \mid X[t_0,t_1]^{\mathrm{c}} = X^{\mathrm{fix}}[t_0,t_1]^{\mathrm{c}}) \geqslant \frac{\delta I_{Y,\ell}}{4} - \frac{1}{2}\,.$$

Proof. The lemma is trivially true when $I_{Y,\ell} = 0$, so suppose that $I_{Y,\ell} = \ell$. Then $Y[0\,..\,(2\ell-1)]$ is retrorse, which implies that at most four distinct values of $X[t_0\,..\,t_1]$ yield the same value of $Z[(t_1+1)\,..\,t_2]$. There are q^ℓ possible values of $X[t_0\,..\,t_1]$, each with the same probability, hence, from the definition of entropy,

$$H(Z[(t_1+1)\,..\,t_2] \mid X[t_0,t_1]^{\mathrm{c}} = X^{\mathrm{fix}}[t_0,t_1]^{\mathrm{c}}) \geqslant \frac{q^\ell}{4} \cdot \frac{1}{q^\ell} \cdot \log_2\left(\frac{1}{4/q^\ell}\right) = \frac{\delta\ell}{4} - \frac{1}{2}\,. \qquad \square$$

Corollary 15. *For any fixed number Y, two intervals $[t_0, t_1]$ and $[t_1 + 1, t_2]$ of the same length ℓ, and any algorithm solving the online multiplication problem on X chosen uniformly at random from $[q^n]$,*

$$\mathbb{E}\,[\,|IT(t_0,t_1,t_2)|\,] \geqslant \frac{\delta I_{Y,\ell}}{8w} - 1\,.$$

We take the same approach as in Section 2.3 and use a lower-bound tree $\mathcal{T}$ with n leaves to obtain the next lemma. The proof is identical to the proof of Lemma 7, only that we use Corollary 15 instead of Corollary 6.

To avoid technicalities we will assume that n and δ are both powers of two and we let the base $q = 2^\delta$.

Lemma 16. *For any fixed number Y and any algorithm solving the online multiplication problem, the expected running time of the algorithm with the number X chosen uniformly at random from $[q^n]$ is at least*

$$\frac{\delta}{8w} \sum_{v \in \mathcal{T}} I_{Y,L(v)} - (n-1)\,.$$

Before giving the proof of Theorem 12, we bound the value of $I_{Y,\ell}$ for both a random number Y and $Y = K_{q,n}$. In order to do so, we will use the following two results by Paterson, Fischer and Meyer [PFM74] which apply to binary numbers. The lemmas are stated in our notation, but the translation from the original notation of [PFM74] is straightforward.

Lemma 17 (Lemma 1 of [PFM74])**.** *For the base $q = 2$ and fixed values of t_0, ℓ, n and $X[t_0, t_1]^{\mathrm{c}}$ (where $t_1 = t_0 + \ell - 1$), such that ℓ is a power of two, each value of Z' can arise from at most two values of X' when $Y = K_{2,n}$.*

Lemma 18 (Corollary of Lemma 5 in [PFM74])**.** *For the base $q = 2$ and fixed values of t_0, ℓ, n and $X[t_0, t_1]^{\mathrm{c}}$ (where $t_1 = t_0 + \ell - 1$), such that ℓ is a power of two, at least half of all possible values of Y' have the property that each value of Z' can arise from at most four different values of X'.*

Lemma 19. *If ℓ is a power of two, then for a random $Y \in [q^n]$, $\mathbb{E}[I_{Y,\ell}] \geqslant \ell/2$, and for $Y = K_{q,n}$, $I_{Y,\ell} = \ell$.*

Proof. Suppose first that $Y = K_{q,n}$. Let ℓ be a power of two and t_0 a non-negative integer. We define X', Y' and Z' as before (see Figure 2). Instead of writing the numbers in base q, we consider their binary expansions, in which each digit is represented by $\delta = \log_2 q$ bits. In binary, we can write X, Y and Z as in Figure 2 if we replace n, t_0 and ℓ with δn, δt_0 and $\delta \ell$, respectively. Note that $\delta\ell$ is a power of two. Since $K_{q,n} = K_{2,\delta n}$, it follows immediately from Lemma 17 that Y' is retrorse and hence $I_{Y,\ell} = \ell$.

Suppose now that Y is chosen uniformly at random from $[q^n]$, hence Y' is a random number in $[q^{2\ell}]$. From Lemma 18 it follows that Y' is retrorse with probability at least a half. Thus, $\mathbb{E}[I_{Y,\ell}] \geqslant \ell/2$. □

Proof of Theorem 12. We assume that n is a power of two. Let Y be a random number in $[q^n]$, either under the uniform distribution or the distribution in which $K_{q,n}$ has probability one and every other number has probability zero. A lower bound on the running time is obtained by taking the expectation of the bound in the statement of Lemma 16. Using linearity of expectation and applying Lemma 19 and Fact 9 finish the proof. Note from Lemma 19 that the expected value $\mathbb{E}[I_{Y,\ell}] = \ell$ when $Y = K_{q,n}$. □

4 Acknowledgements

We are grateful to Mihai Pătraşcu for suggesting the connection between online lower bounds and the recent cell-probe results for dynamic data structures and for very helpful discussions on the topic. We would also like to thank Kasper Green Larsen for the observation that our lower bounds are in fact tight within the cell-probe model. MJ was supported by the EPSRC.

References

[CEPP11] R. Clifford, K. Efremenko, B. Porat, and E. Porat. "A Black Box for Online Approximate Pattern Matching". In: *Information and Computation* 209.4 (2011), pp. 731–736.

[CS11] R. Clifford and B. Sach. "Pattern Matching in Pseudo Real-Time". In: *Journal of Discrete Algorithms* 9.1 (2011), pp. 67–81.

[Day60] D. E. Daykin. "Distribution of bordered persymmetric matrices in a finite field". In: *Journal für die reine und angewandte Mathematik* 203 (1960), pp. 47–54.

[Fre78] M. Fredman. "Observations on the complexity of generating Quasi-Gray codes". In: *SIAM Journal on Computing* 7.2 (1978), pp. 134–146.

[FS73] M. J. Fischer and L. J. Stockmeyer. "Fast On-Line Integer Multiplication". In: *STOC '79: Proc. 5th Annual ACM Symp. Theory of Computing*, pp. 67–72.

[FS89] M. Fredman and M. Saks. "The cell probe complexity of dynamic data structures". In: *STOC '89: Proc. 21st Annual ACM Symp. Theory of Computing*, pp. 345–354.

[Gal81] Z. Galil. "String Matching in Real Time." In: *Journal of the ACM* 28.1 (1981), pp. 134–149.

[Hag98] T. Hagerup. "Sorting and searching on the word RAM". In: *STACS '98: Proc. 15th Annual Symp. on Theoretical Aspects of Computer Science*, pp. 366–398.

[KL96] E. Kaltofen and A. Lobo. "On rank properties of Toeplitz matrices over finite fields". In: *ISSAC '96: 1996 International Symp. on Symbolic and Algebraic computation*, pp. 241–249.

[Mor73] J. Morgenstern. "Note on a Lower Bound on the Linear Complexity of the Fast Fourier Transform". In: *Journal of the ACM* 20.2 (1973), pp. 305–306.

[MP69] M. Minsky and S. Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, 1969.

[MP88] M. Minsky and S. Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, 1988.

[Pan86] V. Y. Pan. "The trade-off between the additive complexity and the asynchronicity of linear and bilinear algorithms". In: *Information Processing Letters* 22.1 (1986), pp. 11 –14.

[Pap79] C. H. Papadimitriou. "Optimality of the Fast Fourier transform". In: *Journal of the ACM* 26 (1 1979), pp. 95–102.

[PD04] M. Pătraşcu and E. D. Demaine. "Tight bounds for the partial-sums problem". In: *SODA '04: Proc. 15th ACM-SIAM Symp. on Discrete Algorithms*, pp. 20–29.

[PD06] M. Pătraşcu and E. D. Demaine. "Logarithmic Lower Bounds in the Cell-Probe Model". In: *SIAM Journal on Computing* 35.4 (2006), pp. 932–963.

[PFM74] M. S. Paterson, M. J. Fischer, and A. R. Meyer. "An Improved Overlap Argument for On-Line Multiplication". In: *SIAM-AMS Proceedings*. Vol. 7. Amer. Math. Soc., pp. 97–111.

[Păt08] M. Pătraşcu. "Lower bound techniques for data structures". PhD thesis. MIT, 2008.

[Yao77] A. C.-C. Yao. "Probabilistic computations: Toward a unified measure of complexity". In: *FOCS '77: Proc. 18th Annual Symp. Foundations of Computer Science*, pp. 222–227.

[Yao81] A. C.-C. Yao. "Should Tables Be Sorted?" In: *Journal of the ACM* 28.3 (1981), pp. 615–628.