A lower bound for the parallel complexity of periodicity on multi dimensional arrays

Clive N. Galley*
Department of Computer Science
Kings College London
University of London

ABSTRACT. Given that an array $A[i_1, \dots, i_d], 1 \le i_1 \le m, \dots 1 \le i_d \le m$ has *a period* $A[p_1, \dots, p_d]$ of dimension $p_1 \times \dots p_d$ if $A[i_1, \dots i_d] = A[i_1 + p_1, \dots, i_d + p_d]$ for $i_1, \dots i_d = 1 \dots m - (p_1, \dots, p_d)$. *The period* of the array is $A[p_1, \dots, p_d]$ for the shortest such $q = p_1, \dots p_d$.

Consider this array A; we prove a lower bound on the computation of the period length less than $m^d/2^d$, of A with time complexity

$$\Omega(\log \log_a m) \ , \ a = m^{d^2}$$

for $O(m^d)$ work on the CRCW PRAM model of computation.

1 Introduction

An array $A[i_1, \dots, i_d], 1 \le i_1 \le m, \dots 1 \le i_d \le m$ hasa *period* $A[p_1, \dots, p_d]$ of dimension $p_1 \times \dots p_d$ if $A[i_1, \dots i_d] = A[i_1+p_1, \dots, i_d+p_d]$ for $i_1, \dots i_d = 1 \dots m - (p_1, \dots, p_d)$. *The period* of the array is $A[p_1, \dots, p_d]$ for the shortest such $q = p_1, \dots p_d$. This paper concentrates on the inter-related algorithmic and combinatorial problem of d dimensional periodicity arising in applications which involve extensive manipulations of arrays and strings. Such application areas are pattern recognition and computer vision, speech synthesis and recognition, data compression and encoding, data communication, text retrieval and editing, etc. Perhaps less typical, but equally important applications are found in graph problems and in molecular biology.

*The author was supported by the 91809000 Engineering and Physical Sciences Research Council Postgraduate Studentship. The author was also partially supported by a University of London Central Research Fund grant.

The primary reason that molecular biology is of great interest to computer scientists is that genes, chromosomes, genomes, proteins, and enzymes can be viewed, at one level, as simply strings (or sequences) of symbols from a finite alphabet (the alphabet C,G,A,T of the four nucleotides, or the alphabet of 20 amino acids). A second reason for computer scientists' interest is that the genome project raises many intractable computational questions of an optimization flavour. While many biologists intuitively realized that a problem such as DNA fragment assembly is intractable and, therefore, developed heuristic algorithms, few proofs of intractability, and even fewer proofs of the quality of the results of the heuristic algorithms have been established. A third reason is that genomic databases are extremely large; for example the human genome, which for one individual consists of 23 chromosomes, consisting of approximately three billion nucleotides. We are facing an information overload that will paralyse our computer systems, unless we develop new techniques, or adapt existing techniques, to summarize and visualize genomic data.

The study in this paper is upon periodicity-related combinatorial problems. These problems were studied over seventy years ago by Thue, see [T1],[T2]. The inequality underlying our bound on periodicity was also stated around that time, in 1917, by Hardy and Ramanujan, see [HR]. This inequality was upon the number of prime factors of any integer n. By considering the bound on primes in [RS] by Rosser and Schoenfield, Breslauer and Galil showed the first lower bound on one dimensional pattern matching, see [BG].

Here we provide a multi dimensional generalisation of the lower bound for periodicity from [BG]. Our bound has an $\Omega(\log\log_a m)$ time complexity, where $a = m^{d^2}$ for an array bounded by m^d, and our bound requires linear work. Given the bound on primes used by an adversary in [BG], an alternative method was used in [GI] using some multiples of primes for the same adversary. In corollary to these multiples of primes we show here that exponents of primes may also be used by the adversary. Moreover these exponents of primes can be shown to take a general form set here to the dimension of the array.

Section 2 contains our proof, followed by a conclusion in Section 3.

2 The Lower Bound

Definition 2.0

A vector $(x_1, \ldots, x_d) \rightarrow (x_{d+1}, \ldots, x_{2d})$ is a prime vector iff

(i) $\{x_1, \ldots, x_{2d}\}$ all force a comparison in one dimension.

(ii) $P \in \{x_1, \ldots, x_{2d}\}$ is such that $\forall\ P\ m/2 < P < m$.

(iii) $P_i \in \{x_i\}, i \in \{1 \ldots d\}$ is such that $m/2 < P_i < 3m/4$.

(iv) $P_j \in \{x_j\}, j \in \{d+1 \ldots 2d\}$ is such that $3m/4 < P_j < m$.

□

We begin with a reminder of the other salient definitions. An array $A[i_1, \ldots, i_d], 1 \leq i_1 \leq m, \ldots 1 \leq i_d \leq m$ has*a period* $A[p_1, \ldots, p_d]$ of dimension $p_1 \times \ldots p_d$ if $A[i_1, \ldots i_d] = A[i_1 + p_1, \ldots, i_d + p_d]$ for $i_1, \ldots i_d = 1 \ldots m - (p_1, \ldots, p_d)$. *The period* of the array is $A[p_1, \ldots, p_d]$ for the shortest such $q = p_1, \ldots p_d$. The array we consider is bounded by m^d, and we use m^d symbol comparisons in each iteration. The lower bound holds for testing whether such an array has a period of length smaller than $\frac{m}{2^d}$.

We use the following strategy, which is a generalization of that used in one dimension by Breslauer and Galil. Given an adversary, he could answer

$$\Omega\left(\log \log_a m\right), \ a = m^{d^2}$$

iterations of vector comparisons in such a way that there is still a choice of *fixing* the input array A in two different ways. One way is that the output array has a period of length smaller than $\frac{m}{2^d}$. The other way is that the output array does not have such a period. This implies that any algorithm using less iterations than this will be erroneous.

We continue with some definitions used in [BG]. An integer k is a possible period length of A if the array A can be fixed consistently using the output from previous vector comparisons in such a way that k is a period length of A. For k to be a period length it is necessary that each residue equivalence class modulo k is fixed to the same symbol. Thus, if $l_1 \ldots l_d = j_1 \ldots j_d$ *mod* k, then $A[l_1] = A[j_1] \ldots A[l_d] = A[j_d]$ where $A[i_d]$ is the i_d-th element of A, and A is a d dimensional array.

At the start of iteration i Breslauer-Galil's adversary will maintain an integer, k_i, which is a possible period length. During iteration i this adversary answers vector comparisons such that some k_{i+1} is also a possible period length, and some symbols of A are fixed. Moreover, given a pattern array bounded by m^d, the following Lemma from the Breslauer-Galil proof is critical to what follows.

Lemma 2.1. ([BG]) *If* $p, q \geq \sqrt{\frac{m}{k_i}}$ *and are relatively prime, then a comparison* $S[l] = S[k]$ *is forced by at most one of* pk_i *and* qk_i. □

We continue with a reminder of the key Lemmata from our improvement to the one dimensional pattern matching lower bound, see [GI].

Lemma 2.2. *Let* $p = p_1 p_2, q = q_1 q_2$, *with*

$$\sqrt{\frac{m}{2k_i}} \leq p_j, q_j \leq \sqrt{\frac{m}{k_i}},$$

where $p_j, q_j, j = 1, 2$ are prime numbers. A comparison is forced by at most one of pk_i and qk_i.

Proof: The lemma holds when $gcd(p, q) = 1$, see [BG].

Assume w.l.o.g. that $gcd(p, q) = p_1 = q_1$, $l \equiv k \bmod pk_i$ and $l \equiv k \bmod qk_i$ for $1 \leq l, k \leq m$. Then we have $l \equiv k \bmod p_2q_2p_1k_i$ and $q_2 > 2$, which implies $p_2q_2p_1k_i > m$; this implies $l = k$, a contradiction. □

Lemma 2.3. *Let $p = p_1p_2p_3, q = q_1q_2q_3$, with*

$$\left(\frac{m}{2k_i}\right)^{1/3} \leq p_j, q_j \leq \left(\frac{m}{k_i}\right)^{1/3}$$

where $p_j, q_j, i = 1, 2, 3$ are prime numbers. A comparison is forced by at most one of pk_i and qk_i.

Proof: The lemma holds when the $gcd(p, q) = 1$, see [BG].

Assume $gcd(p, q) > 1$, $l \equiv k \bmod pk_i$ and $l \equiv k \bmod qk_i$ for $1 \leq l, k \leq m$.

If the $gcd(p, q)$ is just one prime number and w.l.o.g $gcd(p, q) = p_1 = q_1$, then we have $l \equiv k \bmod p_2p_3q_2q_3p_1k_i$ and

$$p_2p_3q_2q_3p_1k_i > m$$

this implies $l = k$, a contradiction.

If the $gcd(p, q)$ is a product of two primes and w.l.o.g let $gcd(p, q) = p_1p_2 = q_1q_2$ then we have $l \equiv k \bmod p_3q_3p_1p_2k_i$ and

$$p_3q_3p_1p_2k_i > m$$

this implies $l = k$, a contradiction. □

There is now a generalisation of the second case from Lemma 2.3 which is needed by the d dimensional lower bound. A generalisation of the first case of Lemma 2.3 would lead to a greater lower bound than that shown here. However there are as many different cases to consider in general, as there are dimensions. We leave such a proof for future work. In [G3] the calculation of a lower bound given this hypothetical generalisation can be found. We give the resulting complexity in the conclusion here.

Lemma 2.4. *For a prime k form all the powers of k from*

$$k^j | k \in \{1 \ldots p\} | j \in \{1 \ldots \log_k p\} | p \in \left\{\frac{m}{2k_i} \cdots \frac{m}{k_i}\right\}$$

A comparison is forced by at most one of two such k's above.

Proof: Let $p = p_1^j$ and $q = q_1^j$ where p_1 and q_1 are two such k's. Assume $gcd(p, q) > 1, l \equiv k \bmod pk_i$ and $l \equiv k \bmod qk_i$, for $1 \leq l, k \leq m$. If $gcd(p, q)$ is the product of two such k^j and wlog let

$$gcd(p, q) = p_1^{j-1} = q_1^{j-1}$$

then we have $l \equiv k \ mod\ p_1^{j-1} p_1 q_1^{j-1}$ and

$$p_1^{j-1} p_1 q_1 k_i = p_1^j q_1 k_i = p q_1 k_i > m, \ \ for\ q_1 > 2$$

This implies $l = k$, a contradiction. □

In the one dimensional case, as above, k_i is a candidate for the period length of a string of length n. However in our many dimensional case which follows, k_i is a candidate for the period length of an n^d array.

We also need the following four theorems from [G] which improve the bound by counting comparison forcing multiples.

Theorem 2.5. [RS] *For $n \geq 17$ the number of prime integers between 1 and n denoted by $\pi(n)$ satisfies the following*

$$\frac{n}{\ln n} \leq \pi(n) \leq \frac{5}{4} \frac{n}{\ln n}$$

□

Corollary 2.6. *For $\log_k n \geq 17$ the number of prime integers between 1 and $\log_k n$ denoted by $\pi(\log_k n)$ satisfies the following*

$$\frac{\log_k n}{\ln \log_k n} \leq \pi(\log_k n) \leq \frac{5}{4} \frac{\log_k n}{\ln \log_k n}$$

Proof: We substitute n by $\log_k n$ in Theorem 2.5 □

Corollary 2.7. *There are at least*

$$\left(\frac{1}{4} \frac{\log_k n}{\ln \log_k n} \right) = q_k$$

distinct prime integers in the range $[\log_k(n/2), \log_k n]$ for $k > 2$.

Proof: By Corollary 2.6 we have

$$\pi(\log_k n) \geq \frac{\log_k n}{\ln \log_k n}$$

and

$$\pi\left(\log_k \frac{n}{2}\right) \leq \frac{5}{4} \frac{\log_k \frac{n}{2}}{\ln \log_k \frac{n}{2}}$$

which implies that

$$\begin{aligned}
\pi(\log_k n) - \pi\left(\log_k \frac{n}{2}\right) &\geq \frac{\log_k n}{\ln \log_k n} - \frac{5}{4} \frac{\log_k \frac{n}{2}}{\ln \log_k \frac{n}{2}} \\
&= \frac{1}{\log k} \left(\frac{\log n}{\ln \log n - \ln \log k} - \frac{5}{4} \frac{\log \frac{n}{2}}{\ln \log \frac{n}{2} - \ln \log k} \right) \\
&> \frac{1}{\log k} \frac{1}{4} \left(\frac{\log n}{\ln \log n - \ln \log k} \right)
\end{aligned} \tag{2.7.1}$$

The corollary follows from (2.7.1) □

Theorem 2.8. *There are at least*

$$\prod_{k=2} \binom{q_k}{O(k)} C \geq k \log n \text{ where } q_k = \frac{1}{4} \frac{\log_k n}{\ln \log_k n}$$

distinct exponents of primes, $\mathcal{P}$, in the range $\frac{n}{2} < \mathcal{P} < n$

Proof: From Corollary 2.7 there are q_k primes in the range $[\log_k(n/2), \log_k n]$. Choosing $k-1$ of these, say $p_2, \ldots, p_k$, will give us an exponent such that

$$\frac{n}{2} \leq 2^{p_2}, \ldots, k^{p_k} \leq n$$

Excluding primes raised to a prime, which are duplicates, we can choose only

$$k - q_k = O(k)$$

(using $\log_k(n/2) \leq p_i \leq \log_k n$), as there are at least q_k such primes by Corollary 2.7. Simplifying gives

$$\prod_{k=2} \left(\frac{q_k^k}{O(k)!}\right) \geq \prod_{k=2} \left(\frac{q_k^k}{O(k^k)}\right) \geq q_k^k \geq k \log n$$

□

In the many dimensional case we apply the same analysis as in [G2], but here we have new prime vectors, and the lower bound, which follows in a similar fashion, requires developing a calculation based on the size of the set of these new prime vectors.

Lemma 2.9. *Given an m^d array, and two arbitrary vectors $\vec{v} = (l_1, \ldots, l_d) \rightarrow (l_{d+1}, \ldots, l_{2d})$, and $\vec{u} = (j_1, \ldots, j_d) \rightarrow (j_{d+1}, \ldots j_{2d})$, $1 \leq l_i \leq m$, and $1 \leq j_i \leq m$, and a prime vector, $\vec{w} = (x_1, \ldots, x_d) \rightarrow (x_{d+1}, \ldots, x_{2d})$, then a vector comparison $\vec{v} = \vec{u}$ is forced by the prime vector $\vec{w}$.*

Proof: Assume $j_1 \equiv l_1 \bmod (x_{d+1})$ and so on until $j_d \equiv l_d \bmod (x_{2d})$.

By d counts of Lemma 2.4, a symbol comparison at $(j_1, \ldots, j_d)$ to $(l_1, \ldots, l_d)$ is forced.

Assume $j_{d+1} \equiv l_{d+1} \bmod (x_1)$ and so on until $j_{2d} \equiv l_{2d} \bmod (x_d)$.

Again by d applications of Lemma 2.4, a symbol comparison at $(j_{d+1}, \ldots, j_{2d})$ to $(l_{d+1}, \ldots, l_{2d})$ is forced.

As both ends of the prime vector, $\vec{w}$, force a symbol comparison at both ends of $\vec{v}$, and $\vec{u}$ a vector comparison is forced. In other words we force the symbol at $(l_1, \ldots, l_d)$ to be compared with the symbol at $(j_1, \ldots j_d)$, and we force the symbol at $(l_{d+1}, \dots, l_{2d})$ to be compared with the symbol at $(j_{d+1}, \dots, j_{2d})$. That is

$$(j_1, \dots, j_d) \rightarrow (j_{d+1}, \dots, j_{2d})$$

is forced by

$$(l_1, \dots, l_d) \rightarrow (l_{d+1}, \dots, l_{2d})$$

The lemma follows □

Lemma 2.10. *There are at least*

$$\frac{1}{4^d}(k \log n)^{2d}$$

prime vectors $\vec{w}$ in the range $n/2 < \vec{w} < n$ for all co-ordinates.

Proof: By Theorem 2.8 there are at least

$$k \log n$$

co-ordinates of distinct integers to choose from such that $n/2 < (p, q) < n$ so there are

$$(k \log n)^d$$

of these co-ordinates in total. To form the vectors we partition according to each dimension giving

$$\frac{1}{4^d}(k \log n)^d$$

thus there are

$$\frac{1}{4^d}(k \log n)^{2d}$$

on consideration of both ends of each prime vector. □

The proof continues in much the same way as in [BG], but the invariants that are involved differ in the following ways:

- k_i is a prime vector multiple as in Lemma 2.9,
- modular residues are in all co-ordinates,
- K_i is the dth power of the function in [BG], that is

$$K_i = \left(m^{1-4^{-(i-1)}}\right)^d$$

Thus an adversary to any algorithm maintains the following invariants at iteration i:

(1) $\frac{1}{2}K_i \leq k_i \leq K_i$

(2) Given an m^d array, then for each symbol fixed to $(l_1, \ldots, l_d) \rightarrow (l_{d+1}, \ldots, l_{2d})$, for every $j_1 \equiv l_1 \bmod (x_{d+1})$, and so on until for every $j_d \equiv l_d \bmod (x_{2d})$, and for every $j_{d+1} \equiv l_{d+1} \bmod (x_1)$, and so on until for every $j_{2d} \equiv l_{2d} \bmod (x_d)$;

$$(j_1, \ldots, j_d) \rightarrow (j_{d+1}, \ldots, j_{2d})$$

is fixed to the same symbol, where $x_1, \ldots, x_{2d}$ are components of a prime vector.

(3) If a symbol comparison is equal then both symbols compared were fixed to the same symbol.

(4) If a symbol comparison is unequal then

  (a) it is between different residues modulo $x_1, x_2, \ldots,$ *and* x_{2d}

  (b) if the symbols were fixed to values then those values are different

(5) The number of fixed symbols satisfies $f_i \leq K_i$.

A candidate denotes a possible new period length after an iteration of any algorithm. Next to be considered are the candidates for k_{i+1}, which are prime vector multiples of k_i, which also, as a check, satisfy the condition of Lemma 2.4 in all co-ordinates. As mentioned earlier this enables us to claim that the one dimensional proof is a special case of our multi dimensional version.

Lemma 2.11. *There are the following candidates for k_{i+1} which are prime vector multiples of k_i, and satisfy the invariant held by the adversary that $\frac{1}{2}K_{i+1} < k_{i+1} < K_{i+1}$, and the pair (k_i, k_{i+1}) satisfy the conditions of Lemma 2.4 :*

$$\frac{1}{4^d}(d \log m)^{2d}$$

(Where we define a vector multiple as the conjunction of the co-ordinate scalar multiples.)

Proof: These candidates are of the form $\vec{r}k_i$ for a prime vector $\vec{r}$. The count of these follows from Lemma 2.10. We set k from Lemma 2.10 to d. □

Lemma 2.12. *Each such candidate satisfies the condition of Lemma 2.4 for all co-ordinates.*

Proof: As $x_1 \ldots x_d k_i \geq K_{i+1}/2$, $x_{d+1} \ldots x_{2d} k_i \geq K_{i+1}$, and $k_i \leq K_i$ then

$$(x_1 \ldots x_d)^d \geq \frac{1}{k_i} \frac{K_{i+1}^d}{2^d K_i^{d-1}} = \frac{m^{d(3d-2) \cdot 4^{-i}}}{2^d k_i} \geq \frac{m^d}{k_i}$$

the cases up to $x_{d+1} \dots x_{2d}$ are all less restrictive (greater) than above. □

Lemma 2.13. *There exists a candidate for k_{i+1} in the range $\frac{1}{2}K_{i+1} \cdots K_{i+1}$ that forces at most*

$$4^d m^{2d} \left(\frac{1}{d \log m}\right)^{2d}$$

vector comparisons. (Recall a vector comparison is defined as the simultaneous comparison of the symbols to be found at the co-ordinates of both ends of the vector.)

Proof: By Lemma 2.10 there are at least

$$\frac{1}{4^d}(d \log m)^{2d}$$

such candidates that are prime vector multiples of k_i. By Lemma 2.8 each of the m^{2d} vector comparisons (m^d for each end of each vector) is forced by at most one of them. □

Theorem 2.14. *Any comparison based algorithm for finding the period length, smaller than $(m/2)^d$, of an array of size*

$$\overbrace{m \times m \cdots m}^{d \text{ times}}$$

using m^d symbol comparisons in each iteration requires

$$\log \log_a m \ , \ a = m^{d^2}$$

iterations.

Proof: The proof is similar to that in [BG] except the invariants hold at iteration $i+1$ as follows :-

The basis for induction follows from the Breslauer-Galil proof, as there are no changes from their proof at that stage. By Lemma 2.13 k_{i+1} exists and it forces at most

$$4^d m^{2d} \left(\frac{1}{d \log m}\right)^{2d}$$

vector comparisons.

These vector comparisons are equal iff $x_1, \dots, x_{2d}$ modulo k_{i+1} fix the residue class modulo k_{i+1} to the same symbol. All others are unequal. k_{i+1} is a prime vector multiple of k_i, so each residue class modulo k_i has $2d$ scalar residue classes modulo $x_1, \dots x_{2d}$ which split into

$$k_{i+1}/x_1, \dots, k_{i+1}/x_{2d}$$

residue classes modulo k_{i+1}. If two indices are in different (respectively the same) residue classes modulo k_i, i.e. all $2d$ scalar residue classes, then they are also in different (respectively the same) all $2d$ scalar residue classes modulo k_{i+1}.

The invariants can now be itemized

(1) By Lemma 2.10, $\frac{1}{2}K_{i+1} < k_{i+1} < K_{i+1}$, as the prime vector multiples are in the required range.

(2) By the argument above, each symbol is still fixed as before.

(3) Likewise, by the argument above, for equal symbols.

(4) (i) Residue classes of any of the $2d$ points may differ to produce an unequal answer.

(ii) Any previous differing class is maintained, by consistency of the calculation. Two multiples which force a comparison composed will not fail to force a comparison.

(5) By induction $f_{i+1} \leq k_{i+1}$. Here we must consider the induction in each co-ordinate in order to validate the proof. Where $f_{i+1}^{x_1}$ is the x_1 co-ordinate of f_{i+1}, and so on until likewise $f_{i+1}^{x_{2d}}$ is the x_{2d} co-ordinate of f_{i+1}, then let the prime vector $x_1 \rightarrow x_{d+1}$ fix $f_{i+1}^{x_1}$, and let the prime vector $x_d \rightarrow x_{2d}$ fix $f_{i+1}^{x_d}$, each having two scalar residue classes. Given that we show in the following proof that $f_{i+1} \leq k_{i+1} \Rightarrow f_{i+1}^{x_1}$, and so on until $f_{i+1}^{x_{2d}} \leq k_{i+1}^{x_1}$, and so on until $k_{i+1}^{x_{2d}}$, and all sets of co-ordinate vectors are of the same size, this implies that $f_{i+1}^{x_1} \leq k_{i+1}^{x_1}$ and so on until $f_{i+1}^{x_{2d}} \leq k_{i+1}^{x_{2d}}$.

Each residue class modulo k_{i+1} has, at most,

$$2^d m^{2d}/k_{i+1} \leq 2 \times 2^d m^{2d}/K_{i+1}$$

elements and

$$f_{i+1} \leq K_i \left[1 + 2 \cdot 2^d 4^d m^{2d} \frac{m^{2d}}{K_{i+1}} \left(\frac{1}{d \log m}\right)^{2d}\right] \leq K_{i+1}$$

$$1 + 2 \cdot 2^d 4^d \frac{m^{4d}}{K_{i+1}} \left(\frac{1}{d \log m}\right)^{2d} \leq \frac{K_{i+1}}{K_i}$$

$$2 \cdot 2^d 4^d m^{4d} \left(\frac{1}{d \log m}\right)^{2d} \leq \left(\frac{K_{i+1}}{K_i} - 1\right) K_{i+1}$$

$$2 \cdot 2^d 4^d m^{3d} \left(\frac{1}{d \log m}\right)^{2d} < m^{d \cdot 4^{-i}} < m^{2d \cdot 4^{-i}} - m^{-d \cdot 4^{-i}}$$

$$4 \cdot 2^{\frac{1}{d}} 2m^3 \left(\frac{1}{d \log m}\right)^2 < m^{4^{-\log\log_a m}} \quad , a = m^{d^2} \quad < m^{4^{-i}}$$

$$f_{i+1}, k_{i+1} < \left(m^{1-4^{-\log\log_a m}}\right)^d \leq \frac{m^d}{2^d}$$

□

3 Conclusion

In conclusion we have the following summary of results

- For three dimensional pattern matching, where one must first compute the period length less than $m^3/8$, we provide a generalisation which improves the constant of the $\Omega(\log\log m)$, linear work, CRCW PRAM lower bound.

- Given a dimension d we prove a lower bound for d dimensional pattern matching, where one must first compute the period length less than $m^d/2^d$, of

  $$\Omega(\log\log_a m) \ , \ a = m^{d^2}$$

  for $O(m^d)$ work on the CRCW PRAM. The bound is subject to the following restriction on m.

  $$\log_d m \geq 17, \ \textit{or} \ m \geq 2^{17^{\log d}}$$

Unlike the bound for two dimensional pattern matching, see [G2], which extends to $O(\log^2 m)$ extra work, our d dimensional bound does not extend to $O(\log^d m)$ extra work. For a more general case than that in Lemma 2.3 of

$$p = p_1 \ldots p_k, q = q_1 \ldots q_k \mid \left(\frac{m}{2k_i}\right)^{1/k} \leq p_j, q_j \leq \left(\frac{m}{k_i}\right)^{1/k}$$

forcing a comparison, a hypothetical lower bound follows of

$$\Omega\left(\log_4 \log_m \left(m^{3d} (\log m)^{2^{2d^2}}\right)^{\frac{1}{6d^3+2d}}\right)$$

time for $O(m^d)$ work, see [G3]. In this case there is a generalisation to $O(m^d \log^d m)$ work.

References

[BG] D. Breslauer and Z. Galil, A lower bound for parallel string matching, *SIAM Journal on Computing* **21** (1992), 856–862.

[G] C. Galley, A greater lower bound for the k functions coarsest set partition problem, submitted to *Parallel Processing Letters* (1996).

[G2] C. Galley, A CRCW PRAM lower bound for problems on two dimensional arrays, submitted to *Journal of Combinatorics Information and Systems Sciences* (1996).

[G3] C. Galley, On the parallel complexity of combinatorial problems, PhD Thesis, Dept of Computer Science, King's College London, University of London, UK, March 1995.

[GI] C. Galley and C. Iliopoulos, On the lower bound for parallel string matching, submitted to *Intern. J. Computer Maths* (1996).

[HR] G. Hardy and S. Ramanujan, The normal number of prime factors of a number n, *Quart. Journ. of Pure and Applied Maths* **48** (1917), 76–92.

[RS] J. Rosser and L. Schoenfield, Approximate formulas for some functions of prime numbers, *Illinois Journal of Mathematics* **6** (1962), 64–94.

[T1] A. Thue, Uber unendliche zeichenreihen, *Norske Vid. Selsk. Skr. Mat. Nat. Kl.* (Kristiana) Nr **7** (1906), 1–22.

[T2] A. Thue, Uber die gegenseitige lage gleicher teile gewisser zeichenreihen, *Norske Vid. Selsk. Skr. Mat. Nat. Kl.* (Kristiana) Nr **1** (1912), 1–67.