Proceedings of the 9th International Conference on Information Technology and Applications in Biomedicine, ITAB 2009, Larnaca, Cyprus, 5-7 November 2009

Mapping uniquely occurring short sequences derived from high throughput technologies to a reference genome

Pavlos Antoniou, Jackie W. Daykin, Costas S. Iliopoulos, Derrick Kourie, Laurent Mouchard and Solon P. Pissis

***Abstract*—Novel high throughput sequencing technology methods have redefined the way genome sequencing is performed. They are able to produce tens of millions of short sequences (reads) in a single experiment and with a much lower cost than previous sequencing methods. Due to this massive amount of data generated by the above systems, efficient algorithms for mapping short sequences to a reference genome are in great demand. In this paper, we present a practical algorithm for addressing the problem of efficiently mapping uniquely occuring short reads to a reference genome. This requires the classification of these short reads into unique and duplicate matches. In particular, we define and solve the *Massive Exact Unique Pattern Matching* problem in genomes.**

Index terms - sequencing, high throughput, short reads, mapping, pattern matching

I. Introduction

High throughput, (or next generation) sequencing technologies have opened new and exiting opportunities in the use of DNA sequences. These new sequencing technologies enable scientists to obtain full genomic sequences for many species in limited time and in an affordable way. They have the potential to assemble a bacterial genome during a single experiment and at a moderate cost [6]. They are able to produce tens of millions of short reads of currently, typically, $25 - 50$ bp in a single run. A major issue for biologists is how to efficiently handle and classify this massive amount of produced data. In particular, an important problem with these technologies, is how to efficiently and accurately map these short reads to a reference genome [9].

The traditional Sanger sequencing methods [15], [16], developed in the mid 70's, have been the workhorse technology for DNA sequencing for almost 30 years. These methods have been slowly replaced by technologies that use different colored fluorescent dyes [14], [18] and polyacrylamide gels. Later, the gels were replaced by capillaries, increasing the length of individually obtained fragments from 450 to 850 bp. Despite the many technological advances, obtaining the complete sequence of a genome is carried out in very large dedicated "sequencing factories", which require hundreds of automated sequencers using highly automated pipelines.

The recent advances in high throughput sequencing technologies and the way they will crucially impact tomorrow's biology have been presented in several articles [5], [11], [17], [19]. The new emerging technologies, which are based on *in vitro* cloning and sequencing-by-synthesis (SBS) [5], offer new perspectives, such as personal genomics, where every individual will be entitled to access their own genome sequence. Foreseen applications are early cancer diagnoses, or tailored medical treatment (adjusted in such a way as to minimize potential side effects by adapting the drug consumption with respect to the individual).

Examples of these new technologies are the Genome Analyzer [1], developed by *Illumina*, which generates millions of very short reads ranging from 25 to 50 bp in a single experiment [6]; the ABI-SOLiD system, which performs massively parallel sequencing by hybridization and ligation [12]; and the Roche-454 system, which generates fewer but longer sequences [7]. The common denominator of these technologies is the fact that they are able to produce a massive amount of relatively short reads. Due to this massive amount of data generated by the above systems, efficient algorithms for mapping short sequences to a reference genome are in great demand.

Popular alignment programs like *BLAST* or *BLAT* are not successful because they focus on the alignment of fewer and longer sequences [7]. Recently, a new thread of applications addressing the short sequences mapping problem has been devised. These applications, (*ELAND* [3], *SeqMap* [7] or *SOAP* [10], to name a few), which are based

Manuscript received July 10, 2009.

Pavlos Antoniou was with King's College, London, UK and is now with the Department of Computer Science at the University of Cyprus, Nicosia, Cyprus (phone: +35722892636; e-mail: panton@cs.ucy.ac.cy).

Jackie W. Daykin, is with the Department of Computer Science, King's College, London, UK (e-mail: jwd@kcl.ac.uk).

Costas S. Iliopoulos is with the Department of Computer Science, King's College, London, UK (e-mail: jwd@kcl.ac.uk).

Derrick Kourie is with the Department of Computer Science, University of Pretoria, Pretoria, South Africa (email: dkourie@cs.up.ac.za).

Laurent Mouchard is with LITIS (EA 4108), at the University of Rouen, 76800 Saint-Etienne-du-Rouvray, France (email: Laurent.Mouchard@univ-rouen.fr).

Solon P. Pissis is with the Department of Computer Science, King's College, London, UK (e-mail: solon.pissis@kcl.ac.uk).

978-1-4244-5379-5/09/$26.00

on the pigeonhole principle [7], make use of indexing and hashing techniques.

In this paper, we define and solve the *Massive Exact Unique Pattern Matching* problem in genomes. In particular, we address the problem of efficiently mapping uniquely occurring short reads to a reference genome. Our approach is based on three key points:

- it preprocesses the genomic sequence based on the reads length, by using word-level parallelism, before mapping the reads to it.
- it does not index and hash the reads, but instead it converts each read to a unique arithmetic value.
- it directly classifies the mapped reads into unique and duplicate matches, i.e. into reads that occur exactly once in the genome and into reads that occur more than once. The uniqueness of a mapped read guarantees an adequate placement on the sequence, and provides anchors that will be used for placing mate-pair reads, as well as other connected reads [9]. It also identifies a substring that is totally region specific, while most of the genome is repetitive.

The rest of the paper is organised as follows. In Section II, the basic definitions that are used throughout the paper are presented. In Section III, the problem solved in this paper is formally defined. Section IV presents the proposed algorithm in detail, including also the complexity analysis. In Section V, we present the experimental results of the proposed algorithm implemented on a real dataset. Finally, we briefly conclude with some future proposals in Section VI.

II. Preliminaries

A *string* is a sequence of zero or more symbols from an alphabet Σ. The set of all strings over Σ is denoted by Σ^* . The length of a string x is denoted by $|x|$. The *empty* string, that is the string of length zero, is denoted by ϵ. The i-th symbol of a string x is denoted by $x[i]$.

A string w is a *substring* of x if $x = uwv$, where $u, v \in \Sigma^*$. We denote by $x[i \ldots j]$ the substring of x that starts at position i and ends at position j. Conversely, x is called a *superstring* of w. A string w is a *prefix* of x if $x = wy$, for $y \in \Sigma^*$. Similarly, w is a *suffix* of x if $x = yw$, for $y \in \Sigma^*$.

In this work, we are considering the finite alphabet Σ for DNA sequences, where $\Sigma = \{A, C, G, T\}$.

III. Problem definition

We denote the short reads generated from high throughput technologies as the set $\rho_0, \rho_1, ..., \rho_{r-1}$ and we call them *patterns*. Notice that r is a very large integer number ($r > 10^7$). Nowadays, the length of the patterns is typically $25 - 50$ bp long. We denote that length as ℓ. We are given a genomic sequence $t = t[1...n]$, where $n > 10^8$. Due to this massive amount of data, specialized solutions are needed for various sequencing-related problems.

A	00
C	01
G	10
T	11

TABLE I
Binary Encoding of DNA alphabet

String x	A	G	C	A	T
Binary form	00	10	01	00	11
Signature $\sigma(x)$	147				

TABLE II
Signature of $AGCAT$

We define the *Massive Exact Unique Pattern Matching* problem in genomes, as follows.

Problem.
Find whether the pattern $\rho_i = \rho_i[1...\ell]$, for all $0 \leq i < r$, with $\rho_i[j] \in \Sigma = \{A, C, G, T\}$, $j \in \{1, ..., \ell\}$, occurs in $t = t[1...n]$, with $t[i] \in \Sigma = \{A, C, G, T\}$, $i \in \{1, ..., n\}$, exactly once.

IV. Massive Exact Unique Pattern Matching

In this section the focus is to find occurrences of pattern ρ_i, for all $0 \leq i < r$, in text $t = t[1...n]$. In particular, we are interested in whether ρ_i occurs in t exactly once.

In order for the procedure to be efficient we will make use of word-level parallelism by compacting strings into single computer words that we call *signatures*. We get the signature $\sigma(x)$ of a string x, by transforming it to its binary equivalent using 2-bits-per-base encoding of the DNA alphabet (see Table I), and packing its decimal value into a computer word (see Table II).

The idea of employing signatures is long known to computer scientists, introduced by Dömölki in [4] in 1964 for his Shift-Or algorithm, a string matching algorithm, which is based on only a few bitwise logical operations. The most well known application is the *four Russians* algorithm, which packs rows of boolean matrices into computer words, hence speeding up boolean matrix multiplication. A randomised version of *fingerprints* (modulo a prime number) was employed by Karp and Rabin in [8] for solving the pattern matching problem. Their method cannot be used in our pattern matching problem as our signatures are small and, thus, there is no practical speed up by reducing them modulo a prime number.

Our aim is to preprocess text t and create two lists, Λ_ℓ and Λ'_ℓ. The first list holds each duplicate substring of length ℓ of t, and the latter holds each unique substring of length ℓ of t.

An outline of the proposed algorithm for solving the Massive Exact Unique Pattern Matching problem is as follows.

i	z_i	L_ℓ[i]
1	GGG	$(1, \sigma(GGG))$
2	GGT	$(2, \sigma(GGT))$
3	GTC	$(3, \sigma(GTC))$
4	TCT	$(4, \sigma(TCT))$
5	CTA	$(5, \sigma(CTA))$

TABLE III
ILLUSTRATION OF STEP 1 FOR $t = GGGTCTA$ AND $\ell = 3$

STEP 1.
We partition the text with the help of a window (sliding window mechanism), which size is ℓ, into a set of substrings $z_1, z_2, ..., z_{n-\ell+1}$, where $z_i = t[i \dots i+\ell-1]$, for all $1 \leq i \leq n-\ell+1$. We compact each substring z_i into the signature $\sigma(z_i)$, pack it into the couple $(i, \sigma(z_i))$, where i represents the starting position of z_i in t, and add the couple to a list L_ℓ. Notice that, as soon as we compact z_1 into $\sigma(z_1)$, then each $\sigma(z_i)$, for all $2 \leq i \leq n-\ell+1$, can be obtained in constant time (using a "shift"-type operation). *Example.* Table III illustrates step 1 for the case of $t = GGGTCTA$ and $\ell = 3$.

STEP 2.
We sort the list L_ℓ based on the signature field $\sigma(z_i)$, using a well-known sorting algorithm e.g. *Quicksort* [2].

STEP 3.
We run sequentially through the sorted list L_ℓ and check whether the signatures in $L_\ell[x]$ and $L_\ell[x+1]$ are equal, for all $1 \leq x \leq n-\ell+1$. If they are equal, then we add $L[x]$ to a new list Λ_ℓ. If not, then $L[x]$ is added to a new list Λ'_ℓ.

STEP 4.
Assuming that the two lists Λ_ℓ and Λ'_ℓ are already created, we compact each pattern ρ_i, for all $0 \leq i < r$, into a signature, in a similar way to step 1. Then, we can determine, by using binary search, whether a pattern ρ_i of length ℓ occurs in t exactly once. If $\sigma(\rho_i) \in \Lambda'_\ell$, then ρ_i is a unique pattern, and the algorithm returns its matching position in t. If $\sigma(\rho_i) \in \Lambda_\ell$, then ρ_i occurs in t more than once. If $\sigma(\rho_i) \notin \Lambda_\ell$ and $\sigma(\rho_i) \notin \Lambda'_\ell$, then ρ_i does not occur in t.

Notice that in the case that $2\ell > w$, where w is the word size of the machine (e.g. 32 or 64 in practice), our algorithm can easily be modified to store the signatures in $\lceil 2\ell/w \rceil$ computer words.

Theorem 1 *Given the text $t = t[1...n]$, the set of patterns $\rho_0, \rho_1, ..., \rho_{r-1}$, and the length of each pattern ℓ, the proposed algorithm solves the Massive Exact Unique Pattern Matching problem in $\mathcal{O}(\lceil \ell/w \rceil (n+r) \log n)$ units of time.*

Proof: In step 1, we create the signatures in $\mathcal{O}(\lceil \ell/w \rceil n)$ time. In step 2, the time required for sorting the list L_ℓ is $\mathcal{O}(\lceil \ell/w \rceil n \log n)$. In step 3, the sequential run through the list L_ℓ takes $\mathcal{O}(\lceil \ell/w \rceil n)$ time. Assuming that the two lists Λ_ℓ and Λ'_ℓ are created, the main step runs in $\mathcal{O}(\lceil \ell/w \rceil r \log n)$ time. Hence, asymptotically, the overall time is $\mathcal{O}(\lceil \ell/w \rceil (n+r) \log n)$, which is $\mathcal{O}((n+r) \log n)$ in practice. ■

Also, the space complexity is $\mathcal{O}(n)$.

V. EXPERIMENTAL RESULTS

In order to evaluate the correctness and efficiency of our algorithm, we have classified and mapped $31,116,663$ Illumina 25 bp reads, taken from RNA-Seq experiments [13], to the mouse chromosome X. We have mapped the reads to genomic sequences of various lengths (4.10^6 - 8.10^6 bp) of the mouse chromosome X, as well as to the whole chromosome sequence ($166,650,296$ bp). The sequence of the mouse chromosome X was retrieved from the *Ensembl* database. Any unknown nucleotides N in the reference genome induce a mismatch with any given nucleotide. The experimental results are illustrated in Table IV. Table V illustrates the experimental results for preprocessing the mouse chromosome X (Steps 1-3).

A direct comparison, in terms of efficiency and sensitivity on mapping, with other mapping programs would not be reliable. The existing well-known mapping programs are designed to allow up to a small number of mismatches. *ELAND*, developed by Solexa, is the fastest known mapping program. It is optimized to map short reads, with length at most 32 bp. As illustrated in [7], it can map $455,384$ out of $11,530,816$ Solexa 25 bp reads to the mouse chromosome X in 345s, allowing up to 2 mismatches. Our experimental results for classifying and mapping $11,530,816$ reads from the same dataset to the mouse chromosome X, allowing no mismatches, are illustrated in Table VI.

The presented experimental results establish both the efficiency of the proposed algorithm, and the sensitivity of our approach in terms of mapping, confirming our theoretical results. Our algorithm can perfectly map $163,756$ out of $11,530,816$ Solexa 25 bp reads to the mouse chromosome X in 63s. Naturally, we expect the number of mapped reads to increase even further when extending the proposed algorithm to allow for a small number of mismatches.

The proposed algorithm was implemented in the ANSI *C* programming language. The program takes as input the length of the short reads, and two files in FASTA format, the first containing the genomic sequence, and the second containing the short reads. The experiments were conducted on a desktop PC with 2GHz Intel CPU and 2 GB memory, running the Linux operating system. The implementation is available at a website (http://www.dcs.kcl.ac.uk/pg/pississo/), which has been set up for maintaining the source code and the documentation.

VI. CONCLUSION

In this paper, we have presented a novel, practical and efficient algorithm to tackle the data emerging from

Genomic sequence length	Running time	Unique	Duplicates	Total
40,000,000 bp	66s	102,432	67,296	169,728
80,000,000 bp	80s	180,416	89,970	270,386
166,650,296 bp	112s	270,133	121,179	391,312

TABLE IV
CLASSIFYING AND MAPPING 31,116,663 ILLUMINA 25 BP READS TO THE MOUSE CHROMOSOME X

Genomic sequence length	Step 1	Step 2	Step 3	Total
40,000,000 bp	1s	7s	0.3s	8.3s
80,000,000 bp	2s	15.6s	0.6s	18.2s
166,650,296 bp	4.3s	36s	1.3s	41.6s

TABLE V
PREPROCESSING THE MOUSE CHROMOSOME X

the new high throughput sequencing technologies. The new technologies produce a huge number of very short sequences and these sequences need to be classified, tagged and recognised as parts of a reference genome. The proposed algorithm can manipulate this data for Massive Exact Unique Pattern matching in genomes.

The presented experimental results are promising, in terms of efficiency and sensitivity on mapping, compared to more traditional approaches, a fact that suggests that further research and development of the method is desirable.

Extending our approach to tackle the Massive Approximate Unique Pattern matching problem is our immediate goal. The idea of using the pigeonhole principle to split each read into several parts can be adopted. By requiring some of the parts (instead of all of them) to be perfectly matched, the non-candidates can be filtered out very quickly [7]. For example, to admit two mismatches, a read can be splitted into four fragments. The two mismatches can exist in at most two of the fragments (at the same time). Then, if we try all six combinations of the two fragments as the seed, we can catch all hits with two mismatches [10]. In addition, due to the massive amount of data, parallelising the presented algorithm is potentially a further target worth investigating.

REFERENCES

[1] S. Bennett. Solexa ltd. *Pharmacogenomics*, 5(4):433–438, 2004.

[2] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, Second Edition*. The MIT Press, September 2001.

[3] A. Cox. (unpublished) *ELAND*: Efficient local alignment of nucleotide data.

Genomic sequence length	Running time	Unique	Duplicates	Total
40,000,000 bp	28s	42,755	29,803	72,558
80,000,000 bp	38s	74,664	39,377	114,041
166,650,296 bp	63s	111,881	51,875	163,756

TABLE VI
CLASSIFYING AND MAPPING 11,530,816 ILLUMINA 25 BP READS TO THE MOUSE CHROMOSOME X

[4] B. Dömölki. An algorithm for syntactic analysis. *Computational Linguistics*, 8:29–46, 1964.

[5] N. Hall. Advanced sequencing technologies and their wider impact in microbiology. *J. Exp. Biol.*, 210:1518–1525, 2007.

[6] D. Hernandez, P. Francois, L. Farinelli, M. Osteras, and J. Schrenzel. *De novo* bacterial genome sequencing: millions of very short reads assembled on a desktop computer. *Genome Res*, March 2008.

[7] H. Jiang and W. H. Wong. Seqmap: mapping massive amount of oligonucleotides to the genome. *Bioinformatics*, 24(20):2395–2396, 2008.

[8] R. M. Karp and M. O. Rabin. Efficient randomized pattern-matching algorithms. *IBM J. Res. Dev.*, 31(2):249–260, 1987.

[9] H. Li, J. Ruan, and R. Durbin. Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Research*, 18(11):1851–1858, 2008.

[10] R. Li, Y. Li, K. Kristiansen, and J. Wang. SOAP: short oligonucleotide alignment program. *Bioinformatics*, 24(5):713–714, 2008.

[11] E. H. Margulies and E. Birney. Approaches to comparative sequence analysis: towards a functional view of vertebrate genomes. *Nat. Rev. Genet.*, 9:303–313, 2008.

[12] O. Morozova and M. A. Marra. Applications of next-generation sequencing technologies in functional genomics. *Genomics*, 92(5):255 – 264, 2008.

[13] A. Mortazavi, B. A. A. Williams, K. McCue, L. Schaeffer, and B. Wold. Mapping and quantifying mammalian transcriptomes by *RNA-Seq*. *Nature methods*, May 2008.

[14] J. M. Prober, G. L. Trainor, R. J. Dam, F. W. Hobbs, C. W. Robertson, R. J. Zagursky, A. J. Cocuzza, M. A. Jensen, and K. Baumeister. A system for rapid DNA sequencing with fluorescent chain-terminating dideoxynucleotides. *Science*, 238:336–341, 1987.

[15] F. Sanger and A. R. Coulson. A rapid method for determining sequences in DNA by primed synthesis with DNA polymerase. *J. Mol. Biol.*, 94:441–448, 1975.

[16] F. Sanger, S. Nicklen, and A. R. Coulson. DNA sequencing with chain-terminating inhibitors. *Proc. Natl. Acad. Sci. USA*, 74:5463–5467, 1977.

[17] S. C. Schuster. Next-generation sequencing transforms today's biology. *Nat. Methods*, 5:16–18, 2008.

[18] L. M. Smith, J. Z. Sanders, R. J. Kaiser, P. Hughes, C. Dodd, C. R. Connell, C. Heiner, S. B. Kent, and L. E. Hood. Fluorescence detection in automated DNA sequence analysis. *Nature*, 321:674–679, 1986.

[19] B. Wold and R. Myers. Sequence consensus methods for functional genomics. *Nature Methods*, 5(1):19–21, 2008.