BIOINFORMATICS ORIGINAL PAPER

Vol. 25 no. 1 2009, pages 6–13
doi:10.1093/bioinformatics/btn565

Sequence analysis

Slider—maximum use of probability information for alignment of short sequence reads and SNP detection

Nawar Malhis[1,*], Yaron S. N. Butterfield[1], Martin Ester[2] and Steven J. M. Jones[1]

[1]Genome Sciences Centre, BC Cancer Agency, Vancouver and [2]School of Computing Science, Simon Fraser University, Burnaby, BC, Canada

Received on July 15, 2008; revised and accepted on October 27, 2008

Advance Access publication October 30, 2008

Associate Editor: Dmitrij Frishman

ABSTRACT

Motivation: A plethora of alignment tools have been created that are designed to best fit different types of alignment conditions. While some of these are made for aligning Illumina Sequence Analyzer reads, none of these are fully utilizing its probability (*prb*) output. In this article, we will introduce a new alignment approach (Slider) that reduces the alignment problem space by utilizing each read base's probabilities given in the *prb* files.

Results: Compared with other aligners, Slider has higher alignment accuracy and efficiency. In addition, given that Slider matches bases with probabilities other than the most probable, it significantly reduces the percentage of base mismatches. The result is that its SNP predictions are more accurate than other SNP prediction approaches used today that start from the most probable sequence, including those using base quality.

Contact: nmalhis@bcgsc.ca

Supplementary information and availability: http://www.bcgsc.ca/platform/bioinfo/software/slider

1 INTRODUCTION

Novel parallel sequencing technologies including sequencing by synthesis (SBS) such as Illumina Sequence Analyzer (for a review, refer to Holt and Jones, 2008), 454 Life Sciences sequencing and Applied Biosystems SOLiD sequencing, coupled with ever decreasing costs, has provided new opportunities to study genomes. This includes the ability to identify sequence aberrations and chromosomal abnormalities, such as single nucleotide polymorphisms (SNPs), insertions, deletions and inversions, on a genome-wide scale. The amount of sequence produced has also provided the computational challenge of developing accurate and reliable software approaches for sequence alignment that can complete within a useful time frame using affordable computer hardware resources.

Here, we focus on the short sequence reads generated by the Illumina Sequence Analyzer G1 and G2 platforms. Each experiment run typically produces in excess of 1 Gb of nucleotide sequence as short reads (ranging from 27 to 42 nt in length). The Illumina sequencing by synthesis technology works by first randomly fragmenting DNA of interest, ligating adapters to each end, and then annealing the DNA fragments to a flow cell surface coated with complementary oligonucleotides. The DNA fragments are then amplified at millions of unique locations across the flow cell surface resulting in a random array of clonal clusters of DNA molecules. Sequencing of the amplified DNA occurs in cycles. At each cycle, all four fluorescently labelled reversibly terminated nucleotides are added to the reaction allowing the single correct nucleotide to be incorporated by the polymerase at each cluster. A subsequent laser excitation allows for four images to be generated each of which capture a specific excited wavelength from each of the four fluorescent labels corresponding to a single base type. Comparing the colour signal intensity of each base for each cluster, probabilities corresponding to the most likely base incorporation are generated and reported in a prb file. The most probable identity of that base is called according to these probabilities resulting in a final read sequence which is reported in a seq file. It is important to note that with each subsequent cycle, the sequencing error rate increases primarily due to the fact that the chemical reactions at each cycle do not complete to 100%, resulting in molecules within in each cluster becoming out of synchronization. This phenomenon, known as 'dephasing' increases the background noise and reduces the accuracy of each base called as the sequencing proceeds.

An important task is to align sequence reads to the appropriate reference genome for the detection of genetic variants. However, the alignment of these short reads presents a number of challenges due to their short lengths and error rate. Traditional methods of sequence alignment have worked well with sequences that are long enough to provide high location specificity whilst tolerating a number of mismatching bases (sequencing errors and/or biological SNPs). For short reads a correct interpretation of the quality of every base call is essential in the correct alignment of reads to the reference and the accurate identification of SNPs. For example, one base-calling error can result in a no alignment match to the reference genome, a match reporting a false SNP, or a misaligned match to a different region of the genome. The quality of the alignment and SNP prediction decreases as the number of inaccurate base calls increase. In general, if a base in the reference sequence has a high enough coverage with a consistent difference from its reference, this base likely represents a polymorphism. This redundant coverage must be high enough to confidently discriminate true variants from sequencing errors, and to account for the fact that different copies of the same read location might be generated from biological aliases (e.g. diploid or mitochondrial chromosomes).

*To whom correspondence should be addressed.

A plethora of alignment tools have been created employing various methods (some of which are described here). A few make use of the quality information from the Illumina *prb* files and these typically use the base probability values to evaluate the probability of the called base only.

In this study, we will introduce a new alignment tool, Slider. Slider is an application for the Illumina Sequence Analyzer output that uses the probability files instead of the sequence files as an input for alignment to a reference sequence or a set of reference sequences. We will show that this approach is more accurate and more efficient than the currently available software packages.

2 METHODS

First, we introduce existing tools and their approach and then describe Slider's algorithm.

2.1 Existing tools

Different nucleotide alignment programs are made to best fit different alignment properties which include: sequence length of references and reads, quality of reads with respect to sequencing errors, sequence similarity to a reference sequence (SNPs, micro-indels), and the number of reads to be aligned with the reference. Some of the more general use aligners are:

- Exonerate (Slater and Birney, 2005): alignment seeds are generated using FSM (Aho and Corasick, 1975) with multiple queries for each single pass by concatenating some reads (Korf and Gish, 2000). High-scoring Segment Pairs (HSPs) are then formed by extending these seeds [in a similar manner as the BLAST algorithm (Altschul *et al.*, 1990)] and finally, these extended HSPs are joined by sparse dynamic programming (Eppstein *et al.*, 1990) to form the alignments.
- MUMmer (Delcher *et al.*, 2002; Kurtz *et al.*, 2004) and MUMmerGPU (Schatz *et al.*, 2007): a memory-resident suffix tree is first generated from the reference sequence(s). Alignment seeds are generated by streaming reads against the reference suffix tree, and then Smith–Waterman alignment is used for tuning.

Given the two input sets, the set of reads and the set of reference sequences, most of the available aligners are based on the following two steps:

1. First, a main memory-resident indexing data structure for either one of the input sets is generated; this indexing facilitates fast random access to that set.
2. The other input set is scanned against this indexed set, alignment seeds are generated, some aligners extend these seeds, and finally a dynamic programming function is used to join seeds that are adjacent and to generate an alignment score taking in consideration read errors, SNPs and/or small insertion or deletions (micro indels).

However, when the aligned sequences have specific properties, building aligners to best fit these properties will reduce the problem space and improve both the efficiency and accuracy of the alignment process. Some aligners are designed especially for aligning the output of Illumina's Sequence Analyzer. These aligners take advantage of the short read lengths for speeding up the alignment (by using hash tables for speeding up seed identification) which allows the processing of a larger number of reads in a reasonable time frame:

- Eland (Illumina): the mapping algorithm which is provided by Illumina as part of its analysis pipeline, divides each read into four parts (A, B, C and D). Given that these reads are short enough, Eland is able to create a memory-resident hash table for this input set of reads parts (quarters). Eland then scans the reference sequences(s) against this hash table; if a read has an exact match for at least two parts within appropriate distance, the other remaining unmatched parts are searched sequentially. Compared with other aligners, ELAND is very efficient; however, Eland can map reads using up to the first 32 bases; for longer sequences the bases after 32 will be mapped sequentially after the alignment of the initial 32.
- RMAP (Smith *et al.*, 2008): in addition to utilizing hash tables for improving the efficiency, RMAP uses quality scores to improve mapping accuracy by not penalizing mismatches for bases with a base-call quality score less than a predetermined cut-off value. Bases of lower quality are considered as wild-cards. To allow up to k mismatches, it divides reads into $k+1$ contiguous seeds, builds a hash table for these seeds, and then scans the reference sequence against this hash table. For each read seed that matches to the reference genome, the entire read is compared with the local area surrounding the location of the matching seed.

As of today, these aligners still have some common disadvantages with respect to the analysis of our target data:

1. Scanning one input set against the other indexed input set requires random access to the indexed set which constitutes very poor space locality and in turn requires the complete index structure to be in main memory. This may not be possible when that input set is large. Failing to locate the index structure in main memory will generate a large amount of swapping (thrashing) and will stall program execution as the indexed input set size increases.
2. By not using all base probability information available, these aligners are required to use a higher level of approximation, such as a higher number of allowed base mismatches (abm). This results in a larger number of reads that are misaligned.
3. By considering only the most probable base, these aligners give a large number of base mismatches that would otherwise be interpretable sequencing errors, resulting in a higher percentage of false positive SNPs. Utilizing quality values of only the most probable bases (Smith *et al.*, 2008), partially reduces these unnecessarily false mismatches.

2.2 Slider

To overcome these problems, it is important to first understand the properties of the Illumina Sequence Analyzer data, and then to build an approach that maximizes the utilization of these properties in order to optimize the alignment (efficiency and accuracy) and SNP prediction accuracy.

In contrast to other aligners, the Slider algorithm uses not only the most probable base, but also all possible bases with a probability above a certain base probability threshold (baseMin*Prb*) provided by the Illumina probability files in order to generate all possible reads with probability above a certain read probability threshold (read_0_Min*Prb*). This reduces the level of alignment approximation needed and improves accuracy (reducing the number misaligned reads and the number of base mismatches). For its core alignment, Slider sorts all these generated reads in lexicographical order and then crosses it sequentially with a presorted table of windows of reference sequence(s) and their reverse complement(s). This approach eliminates the need for an indexed structure by replacing random I/O with sequential I/O which allows Slider to scale to large input datasets. Our test results show that Slider efficiently provides more accurate alignments and SNP calls.

Today, the use of Paired End Tag (PET) data in alignments is an important aspect of extracting more information from these reads. Slider currently processes PET data and we are now expanding on this capability. In this article, we demonstrate Slider's concept of aligning a read which is applicable for both single-end and paired-end tag data.

3 THE SLIDER ALGORITHM

There are three main properties of the Illumina *Sequence Analyzer* output reads that are important to consider:

(1) The number of reads is large; one machine run (a single flow cell) generates about 40 million reads in about 48 h.

(2) These reads are shorter than other sequencers, currently being about 36 nt.

(3) The Sequence Analyzer software provided by Illumina produces probability values associated with each read base. These probability values give at each read location the probabilities of the base to be either A, C, G or T.

The base probability values are provided in the *prb* files in the form of Q-values, where probabilities can then be calculated using:

$$P_{\mathrm{bs}(i)} = 1 - \frac{1}{1+10^{(Q_{\mathrm{bs(i)}}/10)}}$$

where,

Q_{bs} (i): the probability of the base bs, in the form of a Q-value, at location i in a read in the range of $[-Q_{\max}, Q_{\max}]$. Q_{max} is usually set to 40.

P_{bs} (i): the probability of base bs at location i in a read, is in the range of $[P_{\min}, P_{\max}]$.

At a given sequencing cycle, if a cluster is visible in only one of the four images, then the base for that cluster is as well resolved as it can be. We call this base a 'crisp base'. In other words, we define a base as a crisp base if three of its four *prb* values are equal to $-Q_{\max}$, and one is equal to $Q_{\max}$. A crisp read is a read where all the constituent bases are crisp.

For example, when a base is crisp, the base will have such Q and corresponding P-values:

	A	C	G	T
Q	−40	−40	40	−40
P	0.01%	0.01%	99.99%	0.01%

But, if the base is not crisp, the Q-values for a base might look like this:

	A	C	G	T
Q	−40	−20	20	−40
P	0.01%	1%	99%	0.01%

Utilizing these properties, Slider provides major improvements in efficiency and accuracy over other traditional alignment algorithms.

We present some more definitions that will be used in later sections below:

Definitions and notations:

R_{src}: the actual source read of nucleotide sequence.

SZ_{r}: the size (length) of a read.

SZ_{d}: the size (length) of a single database sequence (which is a subsequence of the reference).

R_{ps}: a probable sequence of a read given its *prb* line; crisp reads have only one R_{ps}.

R_{mps}: the most probable sequence (MPS) of a read; this sequence is the one given in the *seq* files which are generally used by many aligners as the only input.

R_{al}: a substring of the reference sequence where an R_{ps} is aligned by an aligner. If R_{al} = R_{src}, we say that the read is accurately aligned. Otherwise, the read is misaligned.

D_{mps}: the edit distance (or the number of different bases) between R_{mps} and R_{src}.

D_{al}: the edit distance between an aligned R_{ps} and R_{al}.

P_{ps}: the probability, 'weight', of an R_{ps}.

$$P_{\mathrm{ps}} = \prod_{i=1}^{SZ_{\mathrm{r}}} P_{\mathrm{bs}(i)}$$

P_{mps}: equals P_{ps} when R_{ps} is R_{mps}. Crisp reads have only one R_{ps}, which is the R_{mps} and this value is about one.

For example, assume a read of 8 bases, with two non-crisp bases (an example of a non-crisp read):

	1	2	3	4	5	6	7	8
Q(A)	−40	−40	−40	40	−40	40	−15	−40
Q(C)	−40	10	−40	−40	−40	−40	−40	−40
Q(G)	40	−10	−40	−40	−40	−40	15	−40
Q(T)	−40	−40	40	−40	40	−40	−40	40

We first convert the Q-values to probabilities:

	1	2	3	4	5	6	7	8
P(A)	0.01	0.01	0.01	99.99	0.01	99.99	3.07	0.01
P(C)	0.01	90.91	0.01	0.01	0.01	0.01	0.01	0.01
P(G)	99.99	9.09	0.01	0.01	0.01	0.01	96.93	0.01
P(T)	0.01	0.01	99.99	0.01	99.99	0.01	0.01	99.99

Then, we generate all possible sequences considering every possible nucleotide with a probability higher than a baseMin*Prb* (a 0.2% value is used by default), and assign a probability for each sequence P_{ps}. Those reads with a cumulative probability or 'weight' less than the read_0_Min*Prb* threshold (a 0.1% value is used by default) are ignored.

	1	2	3	4	5	6	7	8	P_{ps}(%)
Ps/MPS	G	C	T	A	T	A	G	T	87.86
PS	G	G	T	A	T	A	G	T	8.78
PS	G	C	T	A	T	A	A	T	2.78
PS	G	G	T	A	T	A	A	T	0.28

We will loosely define a read specificity as the ability of accurately aligning the read to a unique location on the reference sequence after allowing a limited number of changes on the read from a set of possible changes (S_{ch}). In general, as the size of S_{ch} increases, the read specificity decreases.

Unlike the Roche (454) *Genome Sequencer* 20 where errors are dominated by missing bases (Brockman *et al.*, 2008), errors in Illumina *Sequence Analyzer*, according to our experience, are mostly base miscalls caused by PCR cluster generation (especially in the early stages), cross-talk between clusters (a considerable amount of such errors being generated when the cluster density is high) and optical noise. Most of these base miscalls can be treated according to its probabilities provided in the *prb* files. Each base mismatch in an accurately aligned read can be either a sequencing error or a source mutation (SNP). By using the probability files as input instead of the sequence files, Slider reduces base mismatches for non-crisp bases which eliminates most of the sequencing errors and results in more accurate SNP predictions.

Slider works in five steps:

(1) CreateDB.java for generating a reference database table: this table is generated using a sliding window of size SZ_d, where every subsequence of size SZ_d from the reference and its reverse complement are included. Up to two undefined bases, noted as *N*s in each subsequence, are allowed by substituting all four bases for every *N* and generating all possible subsequences. Finally, this database table is lexicographically sorted. Generating a reference database table needs to be done once for each reference sequence. For example, the human genome reference database table has approximately 6 billion records (counting both forward and reverse complement) that are sorted in lexicographical order. For the human genome, this table was generated in <24 h on a single CPU and used about 160 GB of disk space. Once a reference database table is generated, it can be used for every alignment to that reference, and there is no need for it to be regenerated with each alignment.

(2) CreateSequences.java for generating the P0_Reads Table: using *prb* values, we first generate read sequences R_{ps} for each *prb* line as explained earlier. One unique id is given for all reads that are generated from the same *prb* line source. When a *prb* line has a number of non-crisp bases larger than some threshold value UDmax (a value of 11 is used), only the MPS (most probable sequence) is generated and is marked as a low quality, LQ. LQ sequences will not be used for SNP prediction. The table of all reads generated from all *prb* lines is then lexicographically sorted to form the P0_Reads table.

(3) Alignment.Java: find read locations on the reference sequence with an exact match and one-off match (one base mismatch) to *prb*-derived sequences.

(4) Separation.java: for each reference sequence, aligned reads are then separated into two files: a unique match file and a multiple match file.

(5) SNPsPrediction.java: for each reference sequence, a SNP prediction table, a coverage information file and a set of coverage histogram files are created from the unique matches file of a reference sequence.

Given that our datasets are large (hundreds of millions of records), in order for sorting to not impose main memory limitations on Slider, we used External Merge sorting, which has a runtime complexity of $O(n \log n)$ for *n* records (Knuth, 1998). As each of P0_Reads and P0_Reads tables account for sequencing errors, the size of these tables vary as the sequencing quality changes. The number of records in P0_Reads is about 5–10 times the number of *prb* lines, and each record is 29 bytes, so, if we have an input set of 100 million *prb* lines, P0_reads size is about 15–30 GB. P1_Reads size is mainly determined by the number of reads that did not match the reference and are not LQ. The P1_Reads table has about 10–30 times the number of *prb* lines, so for a 100 million reads input set P1_Reads table is about 30–90 GB. Since mismatches in low probability sequences are less likely to account for real SNPs, by increasing the read_1_MinPrb threshold value, we can reduce the size of the P1_Reads table without significantly affecting SNPs prediction.

While steps 1, 2 and 4 are straightforward, we will now cover in more detail the alignment (3) and the SNPs prediction (5) steps.

3.1 Alignment and efficiency

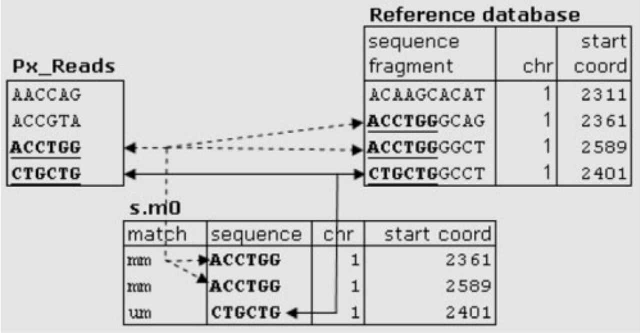
The alignment step is a key feature of Slider. It is fast, accurate and does not have a large memory requirement (a 4 GB machine is enough for the human genome). The slider alignment is based on two steps of exact match alignments:

(1) The first step is to align the P0_Reads table with the reference database table. Given that both tables are lexicographically sorted, two sliding pointers (Figure 1) can do the exact match alignment with each sliding pointer passing once on one of the two tables. If a read from P0_Reads matches more than one database sequence, it is flagged as a multi-match and cannot therefore be unambiguously mapped to the reference genome. Similarly, if reads in P0_Reads with the same id match to different sequences they are also considered as a multi-match. An '.m0' table is generated that holds all matched reads with their matching information (weight, location, unique match versus multiple matches, etc.).

(2) Reads in the P0_Reads table with an id that is not in the '.m0' table and have a weight more than the threshold weight, read_1_Min*Prb*, are used to generate a one-off read table, P1_Reads, by applying all possible one base mutations on these reads. The P1_Reads table is then lexicographically sorted.

(3) Similar to the first step, the P1_Reads table is aligned to the reference database table. A '.m1' table is generated that holds all matched reads.

While some aligners match sequences with up to a small number of mis-matches (typically, this is limited to two as increasing the number of possible mismatches reduces the read specificity), if D_{mps}, the edit distance between R_{src} and R_{mps}, is greater than this value, the read will either fail to align or will misalign. Eland, for example, uses only the most probable sequence to align and others, such as RMAP (Smith *et al.*, 2008) make use of quality information



Reference database

sequence fragment	chr	start coord
ACAAGCACAT	1	2311
ACCTGGGCAG	1	2361
ACCTGGGGCT	1	2589
CTGCTGGCCT	1	2401

Px_Reads

AACCAG
ACCGTA
ACCTGG
CTGCTG

s.m0

match	sequence	chr	start coord
mm	**ACCTGG**	1	2361
mm	**ACCTGG**	1	2589
um	**CTGCTG**	1	2401

Fig. 1. Slider scans both the lexographically sorted reference database and the lexicographically sorted Px_Reads (s.ol0) input table once to generate all exact matches. Exact matches are stored in the sorted s.m0 table. In this example, the set of input sequences is 6 bp (SZ_r = 6), which is aligned to a reference database of 10 bp (SZ_d = 10) oligos created with a sliding window across the reference. Reads that match are indicated in bold and underlined with an example of a unique match indicated by a solid line and that of a multiple match with a dashed line.

(of the most probably base) in a discreet way in determining a degenerate or ‘wild card’ position. However, because the threshold value for non-crisp bases in a read can be large with respect to the number of possible mismatches used by other aligners, this only partially solves the problem.

The analysis of the run time for each aligner must take into consideration a large number of factors including: read size, read quality, reference accuracy and contamination. In general, the computational complexity for both sorting and indexing is of an order of $O(n \ast \log n)$. Let us have an input set of M reads, and a reference sequence of size N. Given that our interest is when both M and N are too large to fit in main memory, the input set of reads are divided into k subsets of size $m = M/k$, such that each of these subset of sequences and their associated parameters can fit in main memory (for most aligners, these subsets are up to 15 million reads). In comparing the computational complexity of sorting with indexing, we can see that:

- Indexing approaches first need to create the indexing structure for each of the k subsets, $O(k \ast m \ast \log m)$, and then to search for every possible subsequence of the reference with the reads’ length in every one of the k-indexed subset of reads $O(k \ast N \ast \log m)$ (Garcia-Molina *et al.*, 2008), since m is constant, then:

  Total: $O(k \ast N + M)$.
- The sorting and streaming needs to sort (external sorting) the input set of reads, $O(M \ast \log M)$, and then stream this sorted reads against a presorted reference $O(N + M)$.

  Total: $O(N + M \ast \log M)$.

This gives the sorting approach an advantage when the input set of reads and the reference genomes are large such as for human.

3.2 SNP prediction

SNP prediction is another key feature of Slider. The goal is to locate places in the reference sequence that are covered with read bases that are not equal to the reference sequence base.

Given that Slider attempts to align most of the possible reads for a *prb* line as a zero-off (U0) match, the final number of Slider’s U0 matches is higher than other aligners, and its number of one-off (U1) matches is much lower. Given that most probability values are considered in the U0 and U1 phases, no U2 alignment is needed for Slider (Table 1). As a result, there are a lower number of base mismatches, and consequently more accurate SNP predictions. This parsimonious approach by Slider means that before a base mismatch is called, the probabilities of all the bases are utilized to ensure that the base is not one of the predicted nucleotides provided by the read *prb* values. Given the large number of sequence reads that are generated by a machine run, it will be expected that many of the actual bases in sequence reads will not be those reported as the most probable.

A reference base can be covered by an aligned read in one of three different categories:

(1) Crisp base match: when the read base is crisp and is equal to the reference base.

(2) Non-crisp base match: when an aligned read base is not crisp and the probability of the nucleotide that is equal to the reference nucleotide is higher than baseMin*Prb*. In this case, it is possible that another base with a non-negligible probability is the actual source base. This would be a putative SNP.

(3) Mismatch bases (no-match): when a base in an aligned read does not match the reference base, i.e. the probability of the base in the *prb* file that matches the reference base is below the baseMin*Prb* threshold; this base is a candidate SNP. Figure 2 shows the probability of a no-match base to be a true SNP as a function of the read weight as found in two datasets.

Mismatch bases are predicted to be SNPs based on two main threshold values:

(1) Coverage: the number of mismatch bases that cover a specific location in the reference sequence. High coverage is needed to overcome the effect of sequencing infidelities.

(2) Percentage: the percentage of mismatch bases at the specific location after ignoring non-crisp base matches.

Table 1. Comparison of number of reads aligned at various lengths between Eland, RMAP and Slider

	CT302, 27 bases			CT302, 32 bases		
	Eland	RMAP	Slider	Eland	RMAP	Slider
U0	1 421 114	1 435 842	1 806 896	1 073 725	1 092 344	1 570 854
U1	431 065	425 641	156 084	527 388	525 338	138 566
U2	178 993	157 701		267 044	259 458	
No. of MB	54 052 593	53 776 925	52 844 376	58 719 548	59 549 564	54 562 874
No. of BMM	789 051	741 043	156 084	1 061 476	1 044 254	138 566
Percentage of BMM	1.44	1.36	0.29	1.78	1.72	0.25

Using the first 27 and 32 for CT302 (a high-coverage control BAC), the numbers of reads that are aligned to its reference control BAC are compared using three aligners, Eland with the MPS input, RMAP with Base Quality input and Slider with *prb* input. The number of reads that are aligned with zero-off (U0), one-off (U1) and two-off (U2) show that Slider aligns a larger number of U0 reads and smaller number of U1 reads than Eland and RMAP. In the last three lines, we can see that the percentage of base mismatches (BMM) calculated by Slider is more than four times smaller than either of Eland or RMAP. We can see that the total number of reads aligned by Slider decreases more than other aligners as the read length increases. This is due to the large increase of LQ reads as the read length increases, and Slider chooses to align only the MPS (U0) of LQ reads, while Eland and RMAP treat LQ reads as regular reads and align them with U0, U1 and U2. This special treatment of LQ reads improves Slider SNPs prediction by reducing the percentage of mismatched bases. Where MB = Matched Bases.

Other factors that affect the accuracy of SNP prediction include:

1. Sequence complexity: we define sequence complexity for a reference sequence with respect to a read size S_{Zr} at any base by the number of unique subsequences with S_{Zr} size in the reference sequence that this base is part of. Considering both the forward and reverse complement, this value can be up to $2 * S_{Zr}$. Bases in a region of higher sequence complexity will less likely be covered with misaligned reads which results in more accurate SNP predictions.
2. Read weight: reads with a higher weight are less likely to misalign, and therefore, mismatches in the alignment of higher weight reads are more likely to be true positive SNPs, Figure 2.

3.2.1 SNPs prediction steps Three counters for each base in the reference sequence are created: cmCount, mCount and nmCount which hold the number of crisp matches, non-crisp matches and non-matches, respectively. Based on these three counts, aligned reads' weight, and reads' base probability, three scores that reflect the likelihood of a SNP at each location in the reference are generated:

- cSNP%: the percentage of non-match base coverage of the total coverage without counting the non-crisp coverage. cSNP% at a location *L* is: cSNP%[*L*] = nmCount[*L*]/(nmCount[*L*] + cmCount[*L*]).
- wSNP%: given that the probability of a mismatch base to be a SNP is positively correlated with its read weight (Fig. 2), wSNP% is a modification of cSNP% that uses the total weight of each read containing that base instead of their counts. Then it is multiplied by the average probability of the predicted SNP nucleotide. wSNP% more accurately predicts the likelihood of SNPs than cSNP%.
- mSNP%: this is a low accuracy score that can be used only when the mCount is high. mSNP% is the average probability of the predicted SNP nucleotide from non-crisp matches for a base at location *L*, for ncSNP% to have a value, a predicted SNP nucleotide is needed, so nmCount must be grater than zero: ncSNP%[*L*] = Avr(PsnpBase[*L*])/(Avr(PsnpBase[*L*]) + Avr(PrefBase[*L*]).

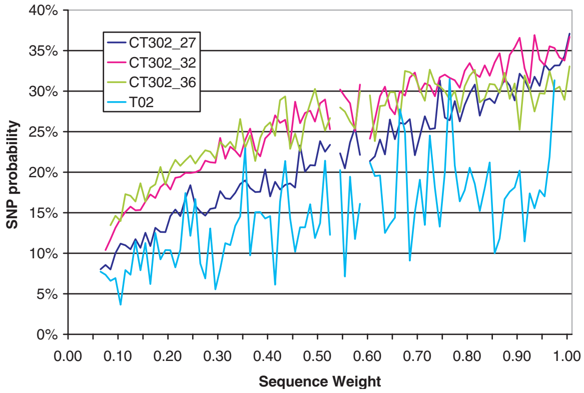


Fig. 2. Probability that a given base mismatch is a true SNP as a function of the read sequence weight.

SNPs are called when the wSNP% and the number of mismatch bases that cover a specific location in the reference sequence are higher than some user provided thresholds.

4 RESULTS

We tested Slider's alignments for both efficiency and accuracy and compared it with competing aligners. In addition to that, SNP prediction of Slider was tested for accuracy using Illumina sequence derived from high-coverage bacterial artificial chromosomes (BACs).

Three different datasets were used to evaluate Slider's performance and compare with other aligners:

1. A high coverage control BAC (CT302).
2. A BAC from a human tumour (T02).
3. One dataset, HT01, which consists of about 82.6 million single-end reads from the human genome.

Two aligners were chosen for comparing results:

1. Eland: as the aligner of choice provided by Illumina, we used Eland as an example of aligners that align the MPSs of reads provided in the seq files.
2. RMAP: we used RMAP to represent the effect of using quality scores to improve mapping accuracy. RMAP uses the *prb* files to reflect the confidence of each base in the MPS.

4.1 Alignment efficiency and scalability

We aligned a set of 12 flowcell lanes, 82.6 million reads in total, to the human reference genome using Eland, RMAP and Slider. Slider aligned the reads for all lanes in 28 h and 30 min with <2 GB of memory (using the –Xmx2G option of the Java VM). This is about two-thirds of the time that Eland took for the same job. RMAP by far took a lot longer to align this dataset (Table 2).

We should mention here that due to their memory requirements, there is a maximum number of reads that can be aligned simultaneously with Eland and RMAP. Therefore, results of aligning for more than one lane are accomplished by aligning to each lane separately and joining their output. Slider does not have such a limitation because it keeps both the input set of reads and the reference sequence stored in files and reads these files sequentially. This reduces Slider memory requirements to a small constant regardless of the data size. Given today's large datasets—such as the Yoruba dataset that has been sequenced by Illumina and consists of 606 lanes (303 PET lanes), with a total of more than 4 billion reads (ftp.ncbi.nih.gov/pub/TraceDB/ShortRead/SRA000271) and those

Table 2. Alignment time comparison

	One lane average alignment time	Total alignment time
Eland	03:38:28	043:41:41
RMAP	95:00:00+	999:59:59+
Slider	05:06:05	028:30:03

For each aligner of Eland, RMAP and Slider, this table show the average lane alignment time in (h:m:s) and the total alignment time for the complete 12 lanes with 82.6 million reads in total.

datasets in the 1000 Genomes project (www.1000genomes.org)—Slider's scalability makes it an attractive choice.

4.2 Alignment accuracy

Starting with the most probable sequence of an edit distance D_{mps} from its actual source R_{src}, most aligners use approximation by introducing all possible x mutations trying to align this sequence to the reference. If the sequence did align with $x < D_{\mathrm{mps}}$, then it will be aligned inaccurately, and x false mismatches are generated. Also, if the location where it was misaligned did have a SNP, this SNP will be covered with a match.

For measuring the alignment accuracy, we aligned a set of reads against both its reference BAC, RefBAC and an extra sequence from a different source, RefEX. Since the RefBAC is the accurate reference for the reads, most reads will be mapped to it; however, if RefEX is large enough, it will have two effects on the alignment:

(1) Some reads which did not align to RefBAC will be misaligned to RefEX. Such misalignments result from a set of factors that we have no control over, such as: sequencing errors, SNPs, indels and/or RefEX size, combined with controllable factors, such as read length, SZ_{r} (longer reads are less likely to be misaligned) and alignment approximation.

(2) Some of the reads that align to RefBAC will also align to RefEX which is due to either sequence similarities in both reference sequences that we have no control over, or because of a misalignment as a result of approximation.

This will make these reads less useful by moving them from the category of unique matches to the category of multiple matches.

As measuring the effect of some of these factors is a complicated issue and beyond the scope of this work, alignments of variable read and reference lengths were used in comparing reads misalignments of Eland, RMAP and Slider.

Using the same set of reads, and the same set of references (RefBAC and RefEX), we can compare the accuracy of different aligners by calculating:

(1) The percentage of reads that are incorrectly uniquely aligned to RefEX, P_{mis}. Where P_{mis} = (number of reads aligned uniquely to RefEX/total number of uniquely aligned reads). The lower P_{mis} is, the higher the aligner accuracy is.

(2) The ratio of reads, P_{uq} that are aligned uniquely to RefBAC when aligning to RefBAC + RefEX to those that are aligned to RefBAC without RefEX. The higher P_{uq} is, the higher the aligner accuracy is.

In a hypothetical ideal case where the target sequence matches exactly the reference sequence, and read sequencing is error free, we will not have any misaligned reads; $P_{\mathrm{mis}} = 0$, and P_{uq} is maximized (multiple matches are only resulted from reference similarities). In reality, the target sequence does not always match the reference sequence and there are sequencing errors. In order to perform the alignment, approximation is required. However, the less approximation we use, the less number of reads will be misaligned but the total number of accurately aligned reads will be reduced also. Different aligners use different approaches for approximation which result in a different number of reads that align and different values for P_{mis}, and P_{uq}.

Table 3. Alignment results

	27		32		36	
	$P_{\mathrm{mis}}(\%)$	$P_{\mathrm{uq}}(\%)$	$P_{\mathrm{mis}}(\%)$	$P_{\mathrm{uq}}(\%)$	$P_{\mathrm{mis}}(\%)$	$P_{\mathrm{uq}}(\%)$
Eland	2.791	76.65	3.002	79.47		
RMAP	2.828	76.69	3.002	79.45	3.520	81.68
Slider	1.169	77.08	1.172	80.19	1.302	83.16

Results of aligning sequences from CT302 to its reference RefBAC and the human genome excluding chromosome 6.

With the use of *prb* files as input and considering all base probability information, Slider is able to significantly reduce approximation compared with other aligners. The result is that less reads are misaligned.

To evaluate Slider's alignment accuracy and compare it with other aligners, alignments were done to their BAC reference sequence and three extra reference sequences RefEX:

- *Caenorhabditis elegans* chromosome I (15M base).
- Human chromosome I (247M base).
- Human genome excluding chromosome 6 because the CT302 BAC is extracted from chromosome 6.

These RefEX sequences are used to measure the level of false alignments for Eland, RMAP and Slider as a function to the RefEX length. Full results are available in the Supplementary Material on the Slider website. In Table 3, values for P_{mis} and P_{uq} for the CT302 BAC with SZ_{r} in {27, 32, 36} aligned against its RefBAC and human genome excluding chromosome 6 (RefEX) shows that the percentage of reads misaligned P_{mis} for each of Eland and RMAP is more than double that of Slider.

4.3 SNPs prediction accuracy

As a result of its low mismatch bases (Section 3.2 and Table 1), Slider is able to give reasonably accurate SNP prediction at low coverage. We measured Slider's SNP prediction accuracy by first taking advantage of the high coverage of CT302 and T02. We generated a list of high-confidence SNPs for each dataset which are easily identified given their high coverage. These lists are then used to evaluate the accuracy of Slider's SNPs prediction by computing the probability of a single base mismatch to reflect a true SNP as a function of its read weight, Figure 2. We can see that mismatches that are in reads with higher weight are more likely to represent actual SNPs. Many factors can affect the probability of a base mismatch to reflect a true SNP including the reference sequence complexity, the length of the read and reference, and the accuracy of the *prb* values. A more complete statistical analysis of SNP prediction accuracy that model these factors is beyond the scope of this article, however, the higher the coverage, the more confident the SNPs call is.

5 DISCUSSION

There are three major advantages of using Slider for the alignment of sequences generated by the Illumina Genome Analyzer and the detection of SNPs based on these alignments:

(1) Higher accuracy in SNPs prediction: by generating a smaller number of base mismatches, Slider's SNPs prediction is

less likely to be confounded by sequence error within the reads.

(2) Smaller number of misalignments: aligners match sequences to a reference using approximation which is done by allowing up to a small number of allowed base mismatches (abm). This abm is needed to align reads with SNPs and sequencing errors. Larger abm values will increase the number of aligned sequences but it will also increase the number of misaligned reads. Slider utilizes the probability values to perform more informed/focused alignments which reduce the need for approximation. For this reason, Slider uses only up to one mismatch base in a read as described in further detail in Section 3.2.

(3) High alignment efficiency and scalability with a low memory requirement: given that neither of the two input datasets (the reference set and the reads set) needs to be in main memory, Slider aligns large sets of reads to large genomes in a short time on a single CPU.

ACKNOWLEDGEMENTS

S.J.M.J. is a senior scholar of the Michael Smith Foundation for Health Research. We thank Anthony Fejes for introducing the initial alignment problem.

Funding: IBM Canada Ltd. (in part).

Conflict of Interest: none declared.

REFERENCES

Aho,A.V. and Corasick,M.J. (1975) Efficient string matching: an aid to bibiographic search. *Commun. ACM*, **18**, 333–340.

Altschul,S.F. *et al.* (1990) Basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410.

Brockman,W. *et al.* (2008) Quality scores and SNP detection in sequencing-by-synthesis systems. *Genome Res.,* **18**, 763–770.

Delcher,A.L. *et al.* (2002) Fast algorithms for large-scale genome alignment and comparison. *Nucleic Acids Res.*, **30**, 2478–2483.

Eppstein,D. *et al.* (1990) Sparse dynamic programming. In *Proceedings 1st Symposium Discrete Algorithms ACM and SIAM*, San Francisco, pp. 513–522.

Garcia-Molina,H. *et al.* (2008) In Index structures, ch. 14. In *Database Systems: The Complete Book*. 2nd edn. Prentice Hall, Upper Saddle River, NJ.

Holt,R.A. and Jones,S.J.M. (2008) The new paradigm of flow cell sequencing. *Genome Res.*, **18**, 839–846.

Knuth,D. (1998) In External sorting, sec 5.4. *The Art of Computer Programming, Vol. 3: Sorting and Searching*. 2nd edn. Addison-Wesley, Reading, Mass.

Korf,I. and Gish,W. (2000) MPBLAST: improved BLAST performance with multiplexed queries. *Bioinformatics*, **16**, 1052–1053.

Kurtz,S. *et al.* (2004) Versatile and open software for comparing large genomes. *Genome Biol.*, **5**, R12.

Schatz,M.C. *et al*. (2007) High-throughput sequence alignment using Graphics Processing Units. *BMC Bioinformatics*, **8**, 474.

Slater,G.S.C. and Birney,E. (2005) Automated generation of heuristics for biological sequence comparison. *BMC Bioinformatics*, **6**, 31.

Smith,A.D. *et al*. (2008) Using quality scores and longer reads improves accuracy of Solexa read mapping. *BMC Bioinformatics*, **9**, 128.