BIOINFORMATICS ORIGINAL PAPER Vol. 27 no. 14 2011, pages 1915–1921
doi:10.1093/bioinformatics/btr303

Sequence analysis

Advance Access publication May 17, 2011

Sensitive and fast mapping of di-base encoded reads

Farhad Hormozdiari[1,†], Faraz Hach[2,†], S. Cenk Sahinalp[2], Evan E. Eichler[1] and Can Alkan[1,*]

[1]Department of Genome Sciences, Howard Hughes Medical Institute, University of Washington, Seattle, WA 98195-5065, USA and [2]School of Computing Science, Simon Fraser University, Burnaby, BC, V5A 1S6, Canada

Associate Editor: Alex Bateman

ABSTRACT

Motivation: Discovering variation among high-throughput sequenced genomes relies on efficient and effective mapping of sequence reads. The speed, sensitivity and accuracy of read mapping are crucial to determining the full spectrum of single nucleotide variants (SNVs) as well as structural variants (SVs) in the donor genomes analyzed.

Results: We present *drFAST*, a read mapper designed for di-base encoded 'color-space' sequences generated with the AB SOLiD platform. *drFAST* is specially designed for better delineation of structural variants, including segmental duplications, and is able to return *all* possible map locations and underlying sequence variation of short reads within a user-specified distance threshold. We show that *drFAST* is more sensitive in comparison to all commonly used aligners such as Bowtie, BFAST and SHRiMP. *drFAST* is also faster than both BFAST and SHRiMP and achieves a mapping speed comparable to Bowtie.

Availability: The source code for *drFAST* is available at http://drfast.sourceforge.net

Contact: calkan@u.washington.edu

Received on March 2, 2011; revised on April 30, 2011; accepted on May 11, 2011

1 INTRODUCTION

Genomic variation between individuals or across species ranges from single nucleotide polymorphisms (SNPs) and structural variation to larger chromosomal rearrangements (Alkan *et al.*, 2011). Thanks to the improvements in sequencing technologies, large-scale genome variation studies such as the 1000 Genomes Project (1000 Genomes Project Consortium, 2010; Mills *et al.*, 2011) have made it possible to better characterize normal human genomic variation and disease (Lupski *et al.*, 2010; Ng *et al.*, 2010; Vissers *et al.*, 2010).

The development of high-throughput sequencing (HTS) technologies has changed the landscape of genome research. The first commercially available HTS technology was from Roche/454 Life Sciences (Margulies *et al.*, 2005) and was used to sequence the genome of James Watson (Wheeler *et al.*, 2008). It was followed by other 'second generation' sequencing platforms that generate orders of magnitude more data for a fraction of the cost, such as Illumina Genome Analyzer (Bentley *et al.*, 2008) and AB SOLiD (McKernan *et al.*, 2009). Third-generation sequencing platforms are now under development, and HeliScope (Pushkarev *et al.*, 2009) and PacBio RS (Eid *et al.*, 2009) were recently made available; however, for the time being, they produce reads with higher error rates.

Analysis of genomic variation using sequencing starts with mapping the randomly sheared and ideally uniformly sampled DNA fragments from the genome. Different properties and error models of sequence reads generated by these technologies require the development of specialized read mapping algorithms for each platform for accurate read alignment and characterization of genomic variants. This becomes more complicated for short reads: due to repeats and duplications in genomes, they can map to multiple locations with equal sequence identity. Leveraging the high sequence coverage and randomly selecting one 'best' location when a read cannot be unambiguously placed has proven to be effective in discovering SNPs and small indels in relatively non-complex areas of the genome (Li *et al.*, 2008a, 2009). However, structural variation detection sensitivity is shown to benefit from tracking all map locations of the reads including suboptimal alignments (Hormozdiari *et al.*, 2009; Lee *et al.*, 2009; Mills *et al.*, 2011), and characterization of segmental duplications is extremely resistant against mapping the reads uniquely (Alkan *et al.*, 2009; Sudmant *et al.*, 2010).

Read mappers can be broadly classified into two categories according to the method used to index the reference genome using either hash tables or suffix arrays [compressed through the Ferragina-Manzini index (Ferragina *et al.*, 2000) with the use of the Burrows-Wheeler Transform (Burrows *et al.*, 1994)]. Hash-based aligners such as MAQ (Li *et al.*, 2008a), SHRiMP (Rumble *et al.*, 2009), mrFAST (Alkan *et al.*, 2009), mrsFAST (Hach *et al.*, 2010) and BFAST (Homer *et al.*, 2009) have poorer performance in comparison to suffix array-based aligners [e.g. BWA (Li *et al.*, 2009), Bowtie (Langmead *et al.*, 2009)] when dealing with short reads; however, their relative performance increases considerably and surpasses the suffix array-based aligners when the read length and thus the number of errors (mismatches or indels) that need to be tolerated increase.

In this article, we describe a hash-based read mapping algorithm named 'di-base read fast alignment search tool' (*drFAST*) designed for the di-base encoded color-space reads generated with the SOLiD platform. The main advantage of di-base encoding is increased base call accuracy due to each base being represented by two 'colors'. This helps in differentiating base calling errors (color-space errors) from real sequence variance, thereby increasing the reliability of detected genomic variants. We show that mapping speed of *drFAST* is higher than other SOLiD-enabled hash-based read mappers, BFAST (Homer *et al.*, 2009) and SHRiMP (Rumble *et al.*, 2009),

*To whom correspondence should be addressed.

†The authors wish it to be known that, in their opinion, the first two authors should be regarded as joint First Authors.

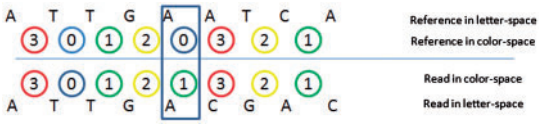

Fig. 1. Translating the read from color-space to letter-space may result in a new sequence different from the original read if there exists a color-space error.

and comparable to suffix array-based aligner Bowtie (Langmead *et al.*, 2009). In addition, *drFAST* was able to map more reads than all the tools we benchmarked. Furthermore, *drFAST* achieves 100% sensitivity if the maximum allowed edit distance is less than L/k, where L is the sequence length and k is the length of the k-mers stored in the hash table (k is set to 12 by default). Coupled with its ability to store all map locations within a user-specified distance threshold and its paired-end (PE) mapping capabilities, *drFAST* can be used to characterize segmental duplications (Alkan *et al.*, 2009; Sudmant *et al.*, 2010) and increase sensitivity of structural variation discovery using VariationHunter (Hormozdiari *et al.*, 2009, 2010).

2 METHODS

For each read sequenced from a donor genome, a mapping algorithm aims to find locations in a 'reference genome' where the read can be aligned exactly or within a small number of errors in the form of substitutions or insertions/deletions (indels). *drFAST* is a read mapper designed for color-space reads generated with the SOLiD platform, and finds 'all' possible map locations for each read of length r in the reference genome within a user-specified e mismatches.

drFAST is a seed-and-extend type algorithm and it builds an index of the reference genome by creating a collision-free hash table for all subsequences of length k (k-mers) of the reference genome. To map the reads, it first partitions each read to $(e+1)$ k-mers and searches for each of these k-mers in the hash table. For each *hit* in the hash table, it then tests if the remainder of the read can be 'extended' by aligning to the reference genome starting at the determined hit location.

How exactly this is done is described below.

2.1 Genome transformation

The sequence data produced with the SOLiD platform are in *color-space* format ($S=\{0,1,2,3\}^*$), where the reference genome sequence is in *letter-space* (i.e. $R=\{A,C,G,T\}^*$). Each color encodes two adjacent base pairs in the read, and each base pair is represented by two colors. Transformation of reads from color-space to letter-space before mapping may result in generating incorrect reads where base call errors exist, as depicted in Figure 1. To avoid such incorrect decoding of reads, we translate the reference genome to color-space and use this transformed genome to create the index.

2.2 Indexing the reference genome

drFAST creates a collision-free hash table for all k-mers in the reference genome. Each entry of this index is a 2-tuple $\tau=(s,L)$, where s is a k-mer from the genome ($k=12$ by default) and L is a list of all positions of the genome starting with this subsequence. The index is maintained in lexicographically sorted order with respect to their subsequences. For a reference genome of length n, the upper bound for the size of its index is $O(n)$; but due to the repetitive nature of genome sequences, the index size is smaller in practice.

2.3 Indexing the reads

drFAST partitions each read of length r into $e+1$ non-overlapping blocks of length k where e is the user-specified maximum Hamming distance allowed for mapping. In the case where $k \leq \lfloor r/(e+1) \rfloor$, the pigeon hole principle guarantees that at least one of these blocks maps to the reference genome with no errors. Similar to the indexing described in Section 2.2, *drFAST* creates an index of blocks computed from all reads in 2-tuples $\tau_r=(s,L_r)$, where L_r denotes the list of reads that include the k-mer s.

2.4 Searching

drFAST compares the reference genome index keys with read index keys to find the locations in the reference genome where a read can be mapped with at most e errors. For each partition of the read, *drFAST* first finds the locations of the reference genome with the identical subsequence (same keys). It then tries to extend the location through sequence alignment of the reads to the genome, and reports those locations where the Hamming distance of the alignment is at most e. A simple loop scans both indices (both are lexicographically sorted); if the keys of the indices are the same (same subsequence) for entries $\tau=(s,L)$ in the reference and $\tau_r=(s,L_r)$ in the read index. Then all entries in L are candidate map locations for each read entry in L_r, and thus the entire list L should be compared with L_r (extending step).

Similar to mrsFAST (Hach *et al.*, 2010), *drFAST* performs 'all-to-all' list comparison using a recursive divide-and-conquer strategy that guarantees cache obliviousness; i.e. asymptotically minimizing the number of costly cache misses (Frigo *et al.*, 1999).

2.5 Extending

The final step is to verify if each read can be aligned to candidate map locations within the user-specified error threshold e. *drFAST* aims to align the color-space read (S_c) to the letter-space sequence (S_l). The aligning process can be considered as finding a letter-space read S_l' that aligns to S_l, and highly similar to S_c if transformed to color-space:

$$\text{argmax}_{S_l'}\left(\text{Sim}(S_l,S_l')+\text{Sim}(S_c,CCG(S_l'))\right) \quad (1)$$

where CCG is the function that transforms the letter-space to color-space as defined by the SOLiD technology, and Sim is the similarity function.

Maximizing the similarity between two sequences is equivalent to minimizing their distance. We use Hamming distance (i.e. the number of mismatches) as the distance measure between two sequences.

$$\text{argmin}_{S_l'}\left(\text{Diff}(S_l,S_l')+\text{Diff}(S_c,\text{CCG}(S_l'))\right) \quad (2)$$

To address the problem, *drFAST* introduces two efficient methods.

2.5.1 Method I: dynamic programming Let $\sum=\{A,C,G,T\}$, and $\sigma,\sigma'\in\sum$, and let $\text{Score}(i,\sigma)$ indicate the optimal alignment of two subsequences $S_l[1..i]$ and $S_c[1..i]$ (from the first to the i-th character) while σ is the last character of S_l'. We then define

$$\text{Score}(i,\sigma)=d(S_l[i],\sigma)+\min_{\sigma'}\{\text{Score}(i-1,\sigma')+d(S_c[i],CCG(\sigma'\sigma))\} \quad (3)$$

where $d(a,b)=1$ if $a\neq b$, and $d(a,b)=0$ otherwise.

The detailed version of Equation (3) is as follows:

$$\text{Score}(i,'\texttt{A}')=d(S_l[i],'\texttt{A}')+\min\begin{cases}\text{Score}(i-1,'\texttt{A}')+d(S_c[i],'0')\\ \text{Score}(i-1,'\texttt{C}')+d(S_c[i],'1')\\ \text{Score}(i-1,'\texttt{G}')+d(S_c[i],'2')\\ \text{Score}(i-1,'\texttt{T}')+d(S_c[i],'3')\end{cases}$$

$$\text{Score}(i,'\texttt{C}')=d(S_l[i],'\texttt{C}')+\min\begin{cases}\text{Score}(i-1,'\texttt{A}')+d(S_c[i],'1')\\ \text{Score}(i-1,'\texttt{C}')+d(S_c[i],'0')\\ \text{Score}(i-1,'\texttt{G}')+d(S_c[i],'3')\\ \text{Score}(i-1,'\texttt{T}')+d(S_c[i],'2')\end{cases}$$

$$\text{Score}(i,'\texttt{G}')=d(S_l[i],'\texttt{G}')+\min\begin{cases}\text{Score}(i-1,'\texttt{A}')+d(S_c[i],'2')\\ \text{Score}(i-1,'\texttt{C}')+d(S_c[i],'3')\\ \text{Score}(i-1,'\texttt{G}')+d(S_c[i],'0')\\ \text{Score}(i-1,'\texttt{T}')+d(S_c[i],'1')\end{cases}$$

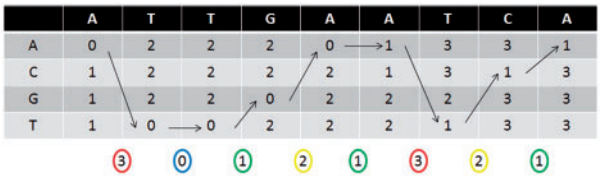

Fig. 2. The dynamic programming table generated to align ATTGAATCA and 30121321 (0 = blue, 1 = green, 2 = yellow, 3 = red). The arrows represent the best alignment between the two sequences.

$$\text{Score}(i,'\text{T}') = d(S_l[i],'\text{T}') + \min \begin{cases} \text{Score}(i-1,'\text{A}') + d(S_c[i],'3') \\ \text{Score}(i-1,'\text{C}') + d(S_c[i],'2') \\ \text{Score}(i-1,'\text{G}') + d(S_c[i],'1') \\ \text{Score}(i-1,'\text{T}') + d(S_c[i],'0') \end{cases}$$

The minimum value of Score($|S_l|,'A'$), Score($|S_l|,'C'$), Score($|S_l|,'G'$) and Score($|S_l|,'T'$) is the Score of the best translation of S_c to S'_l.

Figure 2 shows an example of aligning a letter-space and a color-space sequence using the dynamic programming described in Equation (3). The minimum value in the last column represents the score of the best alignment. Using the backtracking pointer, we can then recover the best alignment sequence.

REMARK 1. *The dynamic programming formulation in Equation (3) will find the optimal solution to the objective function in Equation (2) if the costs of mismatches and read errors are equal to one.*

REMARK 2. *Dynamic programming described in (3) can be modified to handle any cost function for mismatches and read errors.*

Note that the Equation (3) uses Hamming distance but it can be easily generalized for edit distance to allow indels.

$$\text{Score}(i,j,\sigma) = \min \begin{cases} \text{Score}(i-1,j-1,\sigma') + d(S_c[j], CGG(\sigma\sigma')) + d(S_l[i],\sigma) \\ \text{Score}(i-1,j,\sigma') + d(S_l[i],\sigma) \\ \text{Score}(i,j-1,\sigma') + d(S_c[j], CGG(\sigma\sigma')) \end{cases}$$

2.5.2 Method II: transformation-based detection The second method is based on the theoretical design aspect of color-space reads (McKernan *et al.*, 2009). A string of colors $c_1c_2c_3\ldots c_k$ can also be treated as transformations. For example, $C102$ can be written as $f_2(f_0(f_1(C)))$ where the transformation of the colors is applied one after the other. This specific transformation converts C to G, acting as color 3 ($C102 = C3 = G$). For any other base pair, color string 102 will behave exactly as color 3.

The set of color operations is isomorphic to the 'Klein Four Group' (Armstrong, 1988; McKernan *et al.*, 2009). The Klein Four Group is the symmetry group of a rectangle, which has four elements: identity, vertical reflection, horizontal reflection and a 180° rotation. In other words, given the four bases in the corners of a rectangle, each color operation has a one-to-one correspondence with one of the Klein Group elements (Table 1). The Klein Four Group is closed under its elements meaning that if a b are two elements of this group $a \oplus b$ and $b \oplus a$ ($a \oplus b$ means a followed by b) is also an element of the this group. It also has associative, identity, reverse and commutative properties. This means that any sequence of color operations can be considered as one color operation.

We use this property of the color-space reads to detect mismatches. Let two sets of color operations of the same length exist ($c_1\ldots c_k$ and $r_1\ldots r_k$) with different starting color ($c_1 \neq r_2$). For both sets, if any two consecutive colors are replaced with their equivalent (closure property) starting from left-hand side, you will end up with one at the end. If the last color matches with no intermediate matching colors then these two operations show a mismatch of length $k-1$. To illustrate this, consider two color operations 313 and 100. For simplicity, we also consider a leading base C. After applying the color operations, strings *GTA* and *AAA* will be generated, respectively. It can be seen that the last base pair generated using both operations is A and intermediary base pairs are not matching. These two sets of operations have the same transformation. Thus although they generate different sets of base pairs in middle, the final 'product' is the same character.

Table 1. Applying color transformation '3' (left) is the same as applying 180° rotation (right)

A	C
G	T

T	G
C	A

Table 2. Addition table code for strings of colors

$\oplus$	0	1	2	3
0	0	1	2	3
1	1	0	3	2
2	2	3	0	1
3	3	2	1	0

THEOREM 1. *Let $c = c_1c_2c_3c_k$ be a k-color substring of a read aligned with the corresponding color-space reference $r = r_1r_2r_3r_k$. Then c encodes an isolated $(k-1)$-base change if and only if the base position preceding c is not a variant, and the following two equations hold under the Color Addition (Table 2):*

$$\sum_{j=1}^{k} c_j = \sum_{j=1}^{k} r_j$$

For all i from 1 to $k-1$:

$$\sum_{j=1}^{i} c_j \neq \sum_{j=1}^{i} r_j$$

We use Theorem 1 as the basis of our validation function (i.e. extending step). If there is a color mismatch between the read and the reference genome, we consider the next $e+1$ colors to test if there exists any same color transformation of size at most $e+1$ between the read and the genome. Considering a window of limited length, this sometimes may cause incorrect classification of a long stretch of mismatches as two independent read error calls. We refine such calls at the final step.

3 ADDITIONAL FEATURES

Parallelization: an *embarrassingly parallel* wrapper for *drFAST* can be easily written to split the reads into smaller 'chunks' (~1–5 million reads per file) and align on cluster nodes. This approach is the best practice because:

(i) *drFAST* requires <700 MB to load the genome and its index and only a total of ~ 1.3 GB of memory to map 1 million reads to the genome.

(ii) Mapping of each read is independent from mapping the others (except in the case of PE sequences where both ends need to be processed in the same chunk).

PE mapping: SOLiD, like most other HTS technologies, can generate PE sequences. A pair of PE sequences are generated from the prefix and suffix of the same sheared DNA fragment, thus they can be used to increase mapping accuracy and discover structural variation (Alkan *et al.*, 2011; Mills *et al.*, 2011). Current

implementation of *drFAST* supports tracking the PE information, enabling direct use of VariationHunter for structural variation (Hormozdiari *et al.*, 2009) and transposon insertion (Hormozdiari *et al.*, 2010) discovery, as well as NovelSeq (Hajirasouliha *et al.*, 2010) for characterization of novel sequence insertions.

4 RESULTS

To measure the performance of *drFAST*, we compared its two variants to popular color-space read mappers currently available.

Benchmarked software:

- *drFAST*-DP (Dynamic programming variant) (version 0.0.0.2);
- *drFAST*-CT (Color transformation variant) (version 0.0.0.0);
- BFAST (Homer *et al.*, 2009) (version 0.6.4);
- Bowtie (Langmead *et al.*, 2009) (version 0.12.0);
- SHRiMP (Rumble *et al.*, 2009) (version 2.0.1);
- SOCS (Ondov *et al.*, 2008) (version 2.0.3);
- Mapreads (McKernan *et al.*, 2009) (version 2.4.1); and
- PerM (Chen *et al.*, 2009) (version 0.3.3);

Parameters: we used the following parameter settings for these mappers:

- *drFAST*: e=2,3 (error threshold for different runs).
- BFAST: parameters recommended in the BFAST manual.
- Bowtie: n,v = 2,3 (error threshold for different runs); -a (for reporting all); -S (output in SAM format); -C (color-space mapping).
- SHRiMP: -m 1 (score 1 for match); -i -1 (score -1 for mismatch) -x -1 (score -1 for color-space error); -U (ungapped alignment) -o 10 000 (maximum number of alignments for a read); -N 1 (number of threads); -h 96% (≥96% alignment identity).
- SOCS: -x 0 (number of bases to trim); -s 2 (mismatch sensitivity); -t 4 (mismatch tolerance); -m 0 (maximum number of alignments for a read, 0 indicates to report all); -T 1 (number of threads); -l yes (consider the lower case bases in genome).
- Mapreads: S = 0 (color-space mapping); M = 2 (number of mismatches allowed); A = 2 (count adjacent mismatches as one mismatch); Z = 10 000 (maximum number of alignment for a read).
- PerM: –seed S20 (full sensitivity for two SNPs); -v 4 (number of mismatches).

We used the same parameters (for reporting 'all' mapping locations) when available to ensure a fair comparison.

Note that BWA and MAQ are not considered here since they ignore the first two characters of SOLiD reads.

Data, reference genome and computing power: we used both simulated and real datasets for comparisons. We simulated three sets, each with 4 million reads of length 50 bp sampled randomly from chromosome 1 of human reference genome (NCBI build 35) as follows:

- Set 1: we transformed the reads to color-space with no color errors and no mismatches.
- Set 2: reads are transformed with two color errors. To achieve this, we transformed the reads to color-space and then changed the color of two arbitrarily selected non-consecutive positions. Note that if two color errors are consecutive, this might make it impossible to distinguish a read error from a SNP.
- Set 3: generated with no color errors but one SNP.

In addition, we randomly selected 1 million (50 bp long) reads from publicly available color-space reads generated from the genomes of NA18507 (McKernan *et al.*, 2009) (SRX004555), NA10847 (SRX008164) and NA12156. We used the human reference genome (NCBI build 35, unmasked) as the reference genome in all our experiments. The benchmarking results we report are performed on a server with 64 bit Intel Xeon processor and 8 GB of RAM.

Time, accuracy and sensitivity results: we give the comparison results for all the mappers above with respect to the proportion of the reads that have at least one map location on the reference genome (sensitivity), total number of map locations found (comprehensiveness) and time needed to map the reads.

Table 3. Performance results of all tested color-space read aligners on simulated data with error threshold of two mismatches

Dataset	Mapper	Time (min)	Map locations	Reads mapped (%)
Set 1	*drFAST*-DP	65	138715908	**100**
	drFAST-CT	40	137483484	**100**
	BFAST	88	8803840	96.1
	Bowtie v = 2	17	25581176	99.4
	Bowtie n = 2	67	168307651	99.4
	SHRiMP	414	13961155	99.8
	SOCS	45	13357519	**100**
	Mapreads	50	55569848	**100**
	PerM	17	14441796	96.2
Set 2	*drFAST*-DP	42	37652313	**100**
	drFAST-CT	26	36458468	**100**
	BFAST	101	8098581	98.0
	Bowtie v = 2	13	9738234	60.8
	Bowtie n = 2	31	57550920	61.9
	SHRiMP	519	11977512	99.8
	SOCS	90	12909860	**100**
	Mapreads	31	21749155	**100**
	PerM	15	12679070	98
Set 3	*drFAST*-DP	47	76588622	**100**
	drFAST-CT	32	75970911	**100**
	BFAST	105	8982132	97.4
	Bowtie v = 2	16	11030554	49.4
	Bowtie n = 2	43	70508835	51.66
	SHRiMP	472	11859215	99.8
	SOCS	96	9780960	**100**
	Mapreads	37	29799473	**100**
	PerM	15	13140561	97.5

In the case of PerM, we allowed for mapping with up to four mismatches as recommended by its developers, yet its sensitivity failed to reach 100%. Reads are simulated from human reference genome build 35 (chromosome 1). Set 1: no errors; Set 2: color errors; Set 3: substitutions. Values in bold show alignments with 100% sensitivity, higher value implies more sensitivity in the reported alignment.

Table 4. Performance comparison on simulated datasets between *drFAST*-DP, *drFAST*-CT and Bowtie where error threshold is set to three mismatches

Dataset	Mapper	Time (min)	Map locations	Reads mapped (%)
Set 1	*drFAST*-DP	88	364601231	**100**
	drFAST-CT	75	363472241	**100**
	Bowtie v=3	60	56407732	99.4
	Bowtie n=3	101	252735117	99.4
Set 2	*drFAST*-DP	42	118053818	**100**
	drFAST-CT	45	113349741	**100**
	Bowtie v=3	45	23365015	99.4
	Bowtie n=3	50	111931387	99.4
Set 3	*drFAST*-DP	47	215274860	**100**
	drFAST-CT	50	215261940	**100**
	Bowtie v=3	48	28321746	99.4
	Bowtie n=3	75	137015425	99.4

Values in bold show alignments with 100% sensitivity, higher value implies more sensitivity in the reported alignment.

Table 3 shows the results on simulated datasets with error threshold of 2 (color errors and mismatches), except in the case of PerM where we allowed up to four mismatches due to recommendations of its developers. *drFAST* maps *all* the reads from simulated datasets back to the reference genome very efficiently. The closest competitor to *drFAST* appears to be Bowtie, which is, in general, slower than *drFAST*-CT and is not 100% sensitive. Although Bowtie with a parameter setting of v = 2 seems to map each read to more locations than *drFAST*, when no substitutions are present (Set 1), or a single color error is added (Set 2), this is simply due to Bowtie not being stringent on the number of errors it permits disregarding the parameter setting; we noticed that there are mapping locations with more than five color errors.

When the reads involve a nucleotide substitution (Set 3), the number of mapping locations are lower than that of *drFAST*. What is more interesting is the number of reads that can be mapped to the reference genome. It seems like Bowtie can map at most 61.9% of the reads even when they include a single color error (Set 2), in contrast, *drFAST* (both variants) map 100% of the reads. When the errors are in the form of nucleotide substitutions, the proportion of reads mapped by Bowtie drops to 51.66%.

Since Bowtie was the closest competitor to *drFAST*, we performed another experiment on the same datasets by increasing the error threshold to 3 (Table 4). Interestingly for this setting, the proportion of reads mapped by Bowtie is 99.4%, almost matching the 100% mapping sensitivity of *drFAST*. However, both in terms of time and the number of map locations, *drFAST* (both variants) perform better than Bowtie, especially when errors (Set 2 for color errors and Set 3 for nucleotide errors) are present.

As all three sets are generated from chromosome 1 with at most two errors added, a sensitive mapper should be able to map all reads to chromosome 1 when the error threshold is set to 2. In order to experimentally check the accuracy of all locations found by *drFAST*, we simulated the corresponding Illumina reads (letter-space) and aligned to chromosome 1 using mrsFAST. As seen in Table 5 for Sets 1 and 3, *drFAST* finds slightly more mapping locations than mrsFAST, where the sensitivities of both aligners are 100%. The reason *drFAST* can find more mapping locations for SOLiD reads compared with the corresponding Illumina reads is because *drFAST* could map a read like `T0000` to two different positions with base pair contents `TTTT`, and also `CCCC` when one color error is 'corrected'. This is not the case with letter-space reads generated by a platform like Illumina Genome Analyzer. Although it would not be correct to arbitrarily select one 'version' above the other, or returning both alignments as possibilities, we propose to correct such artifacts by incorporating the base pair quality values. This problem will arise only in polyN regions, and thus, we propose to disable error correction of polyN reads where the base quality value of the first base is sufficiently high (i.e. $q > 30$).

Table 5. Number of mapping locations reported by mrsFAST for the same set of simulated reads in letter-space

Dataset	Mapper	Time (min)	Map locations	Reads mapped (%)
Set 1	mrsFAST	20	135 450 193	100
Set 3	mrsFAST	20	75 115 629	100

BFAST and SHRiMP results are not presented for the three real datasets (Table 6) due to: (i) in our experiments, BFAST terminated with error in indexing step; (ii) SHRiMP requires 16 GB of main memory for alignment. Furthermore, both programs are slower than *drFAST* or Bowtie. As a result, we compared *drFAST* with Bowtie, SOCS, Mapreads and PerM with an error threshold of 2 (Table 6) and we compared *drFAST* with Bowtie (closest competitor) with an error threshold of 3 (Table 7). In five out of the six cases, *drFAST* maps significantly more reads, and to substantially more locations, in comparable time. The performance of the two programs are comparable only for NA18507 for n = 2, in terms of mapped reads; however, *drFAST*-CT is slightly faster on this dataset.

We also compared the amount of memory used by each program when mapping 1 million reads to the human reference genome assembly (Table 8).

One important issue to note is that the *drFAST* aligner is aimed at SV/CNV inference and it does not return mapping quality values, which are still essential for accurate SNP detection. However, *drFAST* also returns 'best' map locations for PE and mate pair reads in addition to all possible discordant configurations where 'best' is defined as the mapping with the lowest Hamming distance and with span size closest to the library average. Future releases of *drFAST* will have the capability of returning mapping quality for these best map locations, which will effectively increase the appeal of *drFAST*, and users will be able to use it for both structural variation discovery through multimapping PE and mate pair reads, and SNP discovery. Until this feature is available in *drFAST*, one may need to run other aligners in parallel to discover SNPs.

5 CONCLUSION

This is an exciting time for genomics research. The amount of available (1000 Genomes Project Consortium, 2010) and anticipated (Genome 10K Community of Scientists, 2009) sequence data now arms us to expand our understanding human variation,

Table 6. Performance comparison on real datasets between *drFAST*-DP, *drFAST*-CT, Bowtie, SOCS and PerM on 1 million randomly selected reads from three different sequencing experiments

Dataset	Mapper	Time (min)	Map locations	Reads mapped (%)
	drFAST-DP	114	189276027	36.8
	drFAST-CT	54	149362540	35.6
NA18507	Bowtie v = 2	21	64092233	27.8
	Bowtie n = 2	63	202948323	35.2
	SOCS	320	21081941	35.3
	Mapreads	80	17032680	37.3
	PerM	76	10068062	35.3
	drFAST-DP	200	667928813	47.1
	drFAST-CT	100	512599230	45.9
NA10847	Bowtie v = 2	84	280928112	38.0
	Bowtie n = 2	91	270996634	36.0
	SOCS	420	53668622	44.8
	Mapreads	140	39589079	48.5
	PerM	100	20699652	44.8
	drFAST-DP	136	491158791	33.5
	drFAST-CT	91	440317111	32.5
NA12156	Bowtie v = 2	99	329916108	25.0
	Bowtie n = 2	99	318621596	23.7
	SOCS	400	38246530	31.2
	Mapreads	110	22182469	35.1
	PerM	140	10798496	31.2

We set the error threshold to 2 bp for all aligners, except PerM, where we set the threshold to four as per the PerM developers suggested. We then removed the alignments with more than 2 bp mismatches for comparison purposes.

Table 7. Performance comparison on real datasets between *drFAST*-DP, *drFAST*-CT and Bowtie on 1 million randomly selected reads from three different sequencing experiments. We set the error threshold to 3 bp

Dataset	Mapper	Time (min)	Map locations	Reads mapped (%)
	drFAST-DP	154	309994599	41.5
NA18507	*drFAST*-CT	61	302237779	40.5
	Bowtie v=3	63	145473423	37.1
	Bowtie n=3	78	290357005	36.35
	drFAST-DP	300	1121281408	52.1
NA10847	*drFAST*-CT	141	1092259727	51.6
	Bowtie v=3	182	565114739	47.8
	Bowtie n=3	76	270885799	35.8
	drFAST-DP	310	655648865	42.4
NA12156	*drFAST*-CT	120	639667174	39.5
	Bowtie v=3	187	585191747	34.6
	Bowtie n=3	98	318527434	23.6

disease susceptibility and genome evolution. Although there are inherent accuracy and bias problems associated with different sequencing platforms (Smith *et al.*, 2008), we can also leverage the different 'strengths' of these technologies to increase confidence and comprehensiveness of SNP (Nothnagel *et al.*, 2011) and structural variation (Mills *et al.*, 2011) discovery.

Table 8. Memory required by each software to map 1 million 35 base reads to human reference genome. The memory requirement increases with the number of reads and/or the read length; this increase is typically linear with the increase in the number of base pairs in the dataset

Mapper	Memory usage (GB)
drFAST-DP	1.3
drFAST-CT	1.3
BFAST	≥ 10
Bowtie	4.5
SHRiMP	16
SOCS	≥ 5[a]
PerM	2
Mapread	≥ 8[b]

[a]We set the memory usage of SOCS to 5 GB.
[b]We set the memory usage of Mapreads to 8 GB.

For species where a reference genome is available as in human, mapping sequence reads to this reference assembly is the first step in genome analysis. Sensitivity and accuracy, as well as the speed of read alignment, are crucial for precise characterization of genomic variants. To this end, many mapping algorithms were developed (Alkan *et al.*, 2009; Hach *et al.*, 2010; Homer *et al.*, 2009; Li *et al.*, 2008a, b, 2009; Rumble *et al.*, 2009) focusing mainly on the Illumina Genome Analyzer data, and very little effort was devoted to analyze color-space reads generated with the SOLiD platform (Homer *et al.*, 2009; McKernan *et al.*, 2009; Rumble *et al.*, 2009). The main limitation of the SOLiD-aware read aligners is that they were not optimized for structural variation detection [except for SHRiMP (Rumble *et al.*, 2009), which is more powerful in mapping to more complex areas of the genome], and they are unusable for segmental duplication analysis due to their unique mapping approach (Alkan *et al.*, 2009). On the other hand, by tracking all possible map locations and underlying sequence variation, *drFAST* provides an opportunity to better access and increase 'mappability' in repeat and duplication-rich areas of the genome that are known to harbor much structural variation (Kidd *et al.*, 2008). Although the *sensitivity* of *drFAST* is higher than the other aligners, we also demonstrate *speed enhancements* of both dynamic programming and color transformation versions. Through its readiness to be integrated to VariationHunter (Hormozdiari *et al.*, 2009) for more sensitive SV discovery, to NovelSeq (Hajirasouliha *et al.*, 2010) to characterize novel sequence insertions, and usability for segmental variation detection (Alkan *et al.*, 2009). *drFAST* is an important step forward for recovering additional genetic variation from di-base encoded color-space sequencing.

ACKNOWLEDGEMENTS

E.E.E. is a Howard Hughes Medical Institute Investigator.

Funding: Natural Sciences and Engineering Research Council of Canada (NSERC to S.C.S. in parts); Bioinformatics for Combating Infectious Diseases (BCID to S.C.S. in parts); Michael Smith Foundation for Health Research grants (to S.C.S. in parts); US National Institutes of Health (grants HG004120 and HG005209 to E.E.E.).

Conflict of Interest: E.E.E. is an SAB member of the Pacific Biosciences.

REFERENCES

Alkan,C. *et al*. (2009) Personalized copy number and segmental duplication maps using next-generation sequencing. *Nat. Genet.*, **41**, 1061–1067.

Alkan,C. *et al*. (2011) Genome structural variation discovery and genotyping. *Nat. Rev. Genet*., **12**, 363–376.

Armstrong,M. (1988) *Groups and Symmetry.* Springer, New York, p. 53.

Bentley,D.R. *et al*. (2008) Accurate whole human genome sequencing using reversible terminator chemistry. *Nature*, **456**, 53–59.

Burrows,M. *et al*. (1994) A block sorting lossless data compression algorithm. *Digital Equipment Corporation Technical Report* 124.

Chen,Y. *et al*. (2009) PerM: efficient mapping of short sequencing reads with periodic full sensitive spaced seeds. *Bioinformatics*, **25**, 2514–2521.

Eid,J. *et al*.(2009) Real-time DNA sequencing from single polymerase molecules. *Science*, **323**, 133–138.

Ferragina,P. *et al*. (2000) Opportunistic data structures with applications. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science (FOCS 2000)*, IEEE Computer Society Press, pp. 390–398.

Frigo,M. *et al*. (1999) Cache-oblivious algorithms. In *40th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society, New York, NY, pp. 285–297.

Genome 10K Community of Scientists (2009) Genome 10K: a proposal to obtain whole-genome sequence for 10 000 vertebrate species. *J. Hered.*, **100**, 659–674.

1000 Genomes Project Consortium (2010) A map of human genome variation from population-scale sequencing. *Nature*, **467**, 1061–1073.

Hach,F. *et al*. (2010) mrsFAST: a cache-oblivious algorithm for short-read mapping. *Nat. Methods*, **7**, 576–577.

Hajirasouliha,I. *et al*. (2010) Detection and characterization of novel sequence insertions using paired-end next-generation sequencing. *Bioinformatics*, **26**, 1277–1283.

Homer,N. *et al* (2009) BFAST: an alignment tool for large scale genome resequencing. *PLoS One*, **4**, 12.

Hormozdiari,F. *et al*. (2009) Combinatorial algorithms for structural variation detection in high-throughput sequenced genomes. *Genome Res.*, **19**, 1270–1278.

Hormozdiari,F. *et al*. (2010) Next-generation VariationHunter: combinatorial algorithms for transposon insertion discovery. *Bioinformatics*, **26**, 350–357.

Kidd,J.M. *et al*. (2008) Mapping and sequencing of structural variation from eight human genomes. *Nature*, **453**, 56–64.

Langmead,B. *et al*. (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biol.*, **10**, R25.

Lee,S. *et al*. (2009) MoDIL: detecting small indels from clone-end sequencing with mixtures of distributions. *Nat. Methods*, **6**, 473–474.

Li,H. *et al*. (2008a) Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Res*., **18**, 1851–1858.

Li,R. *et al*. (2008b) SOAP: short oligonucleotide alignment program. *Bioinformatics*, **24**, 713–714.

Li,H. *et al*. (2009) Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*, **25**, 1754–1760.

Lupski,J. *et al*. (2010) Whole-genome sequencing in a patient with Charcot-Marie-Tooth neuropathy. *N. Engl. J. Med.*, **362**, 1181–1191.

Margulies,M. *et al*. (2005) Genome sequencing in microfabricated high-density picolitre reactors. *Nature*, **437**, 376–380.

McKernan,K.J. *et al*. (2009) Sequence and structural variation in a human genome uncovered by short-read, massively parallel ligation sequencing using two-base encoding. *Genome Res.*, **19**, 1527–1541.

Mills,R.E. *et al*. (2011) Mapping copy number variation by population-scale genome sequencing. *Nature*, **470**, 59–65.

Ng,S.B. *et al*. (2010) Exome sequencing identifies mll2 mutations as a cause of Kabuki syndrome. *Nat. Genet.*, **42**, 790–793.

Nothnagel,M. *et al*. (2011) Technology-specific error signatures in the 1000 Genomes Project data. *Hum. Genet* [Epub ahead of print, doi: 10.1007/s00439-011-0971-3, 23 February 2011].

Ondov,B.D. *et al*. (2008) Efficient mapping of Applied Biosystems SOLiD sequence data to a reference genome for functional genomic applications. *Bioinformatics*, **24**, 2776–2777.

Pushkarev,D. *et al*. (2009) Single-molecule sequencing of an individual human genome. *Nat. Biotechnol.*, **27**, 847–850.

Rumble,S.M. *et al*. (2009) SHRiMP: accurate mapping of short color-space reads. *PLoS Comput. Biol.*, **5**, 11.

Smith,D.R. *et al*. (2008) Rapid whole-genome mutational profiling using next-generation sequencing technologies. *Genome Res.*, **18**, 1638–1642.

Sudmant,P.H. *et al*. (2010) Diversity of human copy number variation and multicopy genes. *Science*, **330**, 641–646.

Vissers,L.E. *et al*. (2010) A de novo paradigm for mental retardation. *Nat. Genet.*, **42**, 1109–1112.

Wheeler,D.A. *et al*. (2008) The complete genome of an individual by massively parallel DNA sequencing. *Nature*, **452**, 872–876.