Vo et al. BMC Genomics 2014, 15(Suppl 5):S2
http://www.biomedcentral.com/1471-2164/15/S5/S2

BMC Genomics

RESEARCH Open Access

RandAL: a randomized approach to aligning DNA sequences to reference genomes

Nam S Vo[1], Quang Tran[2], Nobal Niraula[1], Vinhthuy Phan[1*]

From Third IEEE International Conference on Computational Advances in Bio and Medical Sciences (ICCABS 2013)
New Orleans, LA, USA. 12-14 June 2013

Abstract

Background: The alignment of short reads generated by next-generation sequencers to genomes is an important problem in many biomedical and bioinformatics applications. Although many proposed methods work very well on narrow ranges of read lengths, they tend to suffer in performance and alignment quality for reads outside of these ranges.

Results: We introduce RandAL, a novel method that aligns DNA sequences to reference genomes. Our approach utilizes two FM indices to facilitate efficient bidirectional searching, a pruning heuristic to speed up the computing of edit distances, and most importantly, a randomized strategy that enables effective estimation of key parameters. Extensive comparisons showed that RandAL outperformed popular aligners in most instances and was unique in its consistent and accurate performance over a wide range of read lengths and error rates. The software package is publicly available at https://github.com/namsyvo/RandAL.

Conclusions: RandAL promises to align effectively and accurately short reads that come from a variety of technologies with different read lengths and rates of sequencing error.

Background

The alignment of reads to genomes is an important problem in many biomedical applications that relied on next-generation sequencing technologies. This problem is motivated by the fact that genomes for many species have been sequenced. And since one expects genomes within the same species differ little, such "referenced" genomes can facilitate the assembly of new genomes of other individuals within the same species from short reads. To address this problem, researchers have proposed many approaches together with software packages. Nevertheless, sequencing technologies have advanced rapidly, rendering many of these approaches ineffective or inefficient or both. One aspect that continually changes is the read length. Advanced technologies generally produce longer reads (with better accuracy). On the other hand, technologies that produce shorter reads can be less expensive and are therefore attractive in terms of cost. Thus, it is desirable to have algorithms and tools that perform well across different read lengths ranging from 35 to several hundreds basepairs.

Nevertheless, many existing algorithms struggle to perform consistently across a wide range of read lengths. Methods such as Bowtie [1] and Burrows-Wheeler Alignment (BWA) [2] tend to perform better with shorter reads. Bowtie uses the Burrows-Wheeler Transform (BWT) and FM index to build a permanent index of the reference genome. It then applies backtracking algorithm to find alignments. BWA also utilizes the BWT, but unlike Bowtie, can handle gaps and mismatches in the reads. More advanced versions of these methods include Bowtie2 [3] and BWASW [4] which are designed to work with longer reads. Bowtie2 can align reads with gaps and works better than Bowtie at longer reads. BWA-SW exploits the BWT and several heuristics to speed up the local alignment of reads.

* Correspondence: vphan@memphis.edu
[1]Department of Computer Science, University of Memphis, Memphis, TN 38152, USA
Full list of author information is available at the end of the article

Many techniques utilize data structures and techniques such as the BWT, FM index, suffix arrays, suffix trees/tries, hash tables or q-grams [5-11], aiming to speed up substring querrying. Additional heuristics are also used to enhance efficiency. Bowtie2 [3] and CUSHAW2 [12], for example, use *seeds* to quickly identify true candidates for alignment. GASSST [9] uses a filtering technique to reduce noisy seeds. Implementations of some of these approaches, e.g. Bowtie2, CUSHAW2, take advantages of parallelism or special-purpose architectures. The use of heuristics can improve performance several folds, but might lead to over-tuning parameters to a particular set of inputs, e.g. read lengths, species, or base error rates.

We introduce RandAL, an aligner based on a novel algorithm that performs consistently well over a wide range of read lengths, from 35 to several hundreds base pairs. We employ two FM indices for efficient bidirectional (exact) substring matching. To deal with inexact matching (i.e. allowing gaps), first, we find common substrings between reads and the reference genome. Then, these common substrings are extended to complete alignments based on a bounded threshold on the edit distance. We use a special pruning mechanism to shorten vastly the running time of computing edit distances in a vast majority of cases. The use of randomization in aligning reads to genomes increases the probability of finding seeds quickly and enables us determine methodologically important parameters to speed up the entire alignment process. Preliminary results show that our algorithm performed consistently well on a wide range of read lengths across several bacterial and eukaryotic genomes. The alignment quality of our method was better or generally as good as that of all compared methods.

Methods

Given a reference genome $\mathcal{S}$ and a set of reads $\mathbf{R} = \{r_1, ..., r_n\}$, the main problem is to align each r_i to $\mathcal{S}$. The reference genome $\mathcal{S}$ and the reads are DNA sequences, or strings over the alphabet of 4 characters, $\Sigma = \{A, G, C, T\}$. The alignment of a read r to $\mathcal{S}$ is essentially finding a substring of $\mathcal{S}$ that matches r the most. At the moment, we assume that these reads are not *paired-end* reads. The set of reads $\mathbf{R}$ are substrings of another genome $\mathcal{R}$ that is different from, but belongs to the same species as $\mathcal{S}$. By aligning reads in $\mathbf{R}$ to $\mathcal{S}$, we implicitly reconstruct the genome $\mathcal{R}$.

Our strategy for read alignment is based on these ideas:

1. Detection of identical substring matches between r and $\mathcal{S}$ is based on common substrings of r and $\mathcal{S}$. As we know r and $\mathcal{S}$ differ only slightly, we expect long common substrings exist.
2. A special data structure called the FM index is used to facilitate memory-efficient, time-optimal exact string matching. This data structure facilitates efficient detection of long common substrings between r and $\mathcal{S}$.
3. Randomization is employed to find common substrings between r and $\mathcal{S}$ efficiently and *methodologically*. Randomization empowers us to methodologically determine important parameters that are used in critical steps of the algorithm. This translates into consistent performance in terms of time and accuracy across different species.
4. To account for insertion/deletion polymorphisms, we utilize the edit distance to provide an accurate measure for read alignment. Additionally, we employ a pruning heuristic to shorten the computation of edit distance, without com promising quality of alignment.

These ideas will be discussed in greater detail in the following sections.

Indexing the reference genome

Naive string matching takes quadratic time and therefore is too costly. Researchers have used data structures such as suffix tree, suffix array, and FM index to speed up string matching significantly. The FM index [13] in particular is desirable because it allows exact string matching to be done optimally in $O(m)$ time, where m is the length of the query (i.e. the read), and is very space efficient. The FM index of the genome is a substring index that takes advantage of properties of the Burrows-Wheeler transform to search incrementally all suffices of a read in the reference genome. This allows linear time (in read length) searching for exact substring matches. By design, the search direction is in reverse (backward) order with respect to the sequence.

To facilitate bidirectional string matching (to be discussed next), we employ two *FM indices*. A conventional FM index that traces substring matches *backward* is denoted as $\mathfrak{B}$. To facilitate searching in the forward dimension, we created an FM index for the reverse of the reference genome, $\mathcal{S}$. Searching using this index, denoted as $\mathfrak{F}$, is equivalent to search in the forward direction in $\mathcal{S}$. The pair of indices $(\mathfrak{F}, \mathfrak{B})$ helps us identify long identical stretches of DNA in the reference genome $\mathcal{S}$ and each read r_i.

Finding common substrings between reads and genomes

Given a read r and a specific position p in r, Algorithm 1 outlines the steps in finding longest common substrings of r and the reference genome $\mathcal{S}$, *with respect to* p; Figure 1 illustrates the conceptual goal of this algorithm. Longest common substrings (with respect to p) are

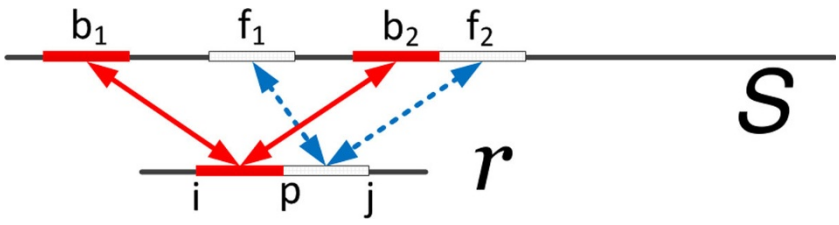

Figure 1 Illustration of Algorithm 1: finding common substring. $r_{i...p-1}$ and $r_{p...j}$ may match several substrings of the genome $\mathcal{S}$, but fews of these (e.g. b_2 and f_2) form contiguous substrings.

constructed by concatenating maximal matches between substrings of $\mathcal{S}$ and those of r, which begin and end at p. Searching for matches between substrings of $\mathcal{S}$ and substrings of r is facilitated by the backward and forward FM indices $\mathfrak{B}$ and $\mathfrak{F}$. To save time and reduce false positives, we only consider common substrings with lengths at least W.

The choice of W is important. If W is too small, M is large, and we will consider many common substrings between the read and the genome to construct alignments between the read and the genome. The more common substrings we consider, the more likely we can find the correct position of the read in the genome to align; but we also more likely make mistakes of aligning the read to an incorrect position. In other words, with smaller W, we might get more true positives (correct alignments) and more false positives (incorrect alignments) at the same time. On the other hand, if W is too large, we might not be able to find any common substrings and consequently unable to align the read to the genome. Therefore, inappropriate choices of W results in bad performance.

Algorithm 1 CommonSubstrings(read *r*, position *p*)

1: Let B be substrings of reference genome $\mathcal{S}$, which match exactly & maximally to $r_{i...p-1}$.
2: Let F be substrings of reference genome $\mathcal{S}$, which match exactly & maximally to $r_{p...j}$.
3: $M := \varnothing$
4: **for** each $b \in B$ **do**
5: **for** each $f \in F$ **do**
6: Let $s := b \oplus f$ be a concatenation of b and f.
7: **if** s is a contiguous block in $\mathcal{S}$ and $|s| \geq W$ **then**
8: $M := M \cup s$
9: **return** M

Our strategy for determining good values of W is based on randomization. As we shall see soon, the value p given to Algorithm 1 would be a random index of the read. To calculate W, first suppose that the correct substring of the reference genome $\mathcal{S}$ to align to the read r is r'. Let d be the edit distance between r and r'. These d mismatches divide r into $d + 1$ blocks. Each block (except the last one) includes the closest mismatch to it. Let the sizes of the blocks be $m_1, m_2, \ldots \ldots \ldots, m_{d+1}$. We have $|r| = m = \sum_{i=1}^{d+1} m_i$.

The random choice of p implies that the common substring found by Algorithm 1 would be a random block, which is selected with probability $p_i = \frac{m_i}{m}$. This implies that the expected size of block i is $E[S_i] = m_i p_i = \frac{m_i^2}{m}$. Thus, the expected size of a random block, i.e. the expected length of the common substring, is $E[X] = \sum_{i=1}^{d+1} E[X_i] = \sum_{i=1}^{d+1} \frac{m_i^2}{m}$.

The Cauche-Schwarz inequality tells us that

$$\left(\sum_{i=1}^{d+1} \frac{1}{d+1}\frac{m_i}{m}\right)^2 \leq \sum_{i=1}^{d+1} \left(\frac{1}{d+1}\right)^2 \sum_{i=1}^{d+1} \left(\frac{m_i}{m}\right)^2$$

After simplifying, these imply that $E[S] \geq \frac{m}{d+1}$. In other words, we have established that:

Lemma: The expected length of the common substring between a read and the reference genome found by Algorithm 1 is at least $\frac{m}{d+1}$.

Although we do not know what d, the distance between r and its aligned substring r', is, it can be estimated by the rates of single nucleotide polymorphism (SNP) of the given genome and given rate of sequencing error. Let b be the rate of each nucleotide being mutated or sequenced erroneously, which we may assume to be distributed by a binomial distribution with mean $\mu = mb$ and variance $\sigma^2 = mb(1 - b)$, where m is the read length.

Although we do not know exactly what d is, its upper bound t might be estimated by $\mu + c\sigma$, for some constant c. With 100,000 reads, we found that $c = 4$ produces good performance with high true positives and low false positives.

In summary, the two critical parameters of our method t and W are methodologically derived as follows:

- The upper bound of the distance between a read and its aligned string, $t = \left\lceil mb + 4\sqrt{mb(1-b)} \right\rceil$.

• The lower bound of the expected length of common substrings, $W \sim \frac{m}{t} \leq \frac{m}{d+1} \leq E[S]$.

W appears in Algorithm 1, and t appears in Algorithm 2, which is the next step after finding common substrings between reads and the reference genome.

Algorithm 2 AlignRead(read r)
1: $p := 1$
2: $m := |r|$
3: **for** i from 1 to A **do**
4: $C := \emptyset$
5: $M :=$ CommonSubstrings(r, p)
6: **for** each $s \in M$, which is a substring of $\mathcal{S}$ **do**
7: Let $r_{i...j}$ be the substring of r that matches s exactly.
8: Let s_L be the $(i-1)$-substring of $\mathcal{S}$, preceding s
9: Let sR be the $(m-j)$-substring of $\mathcal{S}$, following s
10: $d :=$ edit-dist($r_{1...i-1}$, s_L) + edit-dist($r_{j+1...m}$, s_R)
11: **if** $d \leq t$ **then**
12: $C := C \cup (s_L \oplus s \oplus s_R)$
13: **if** C has at least one sequences **then**
14: Return "fail to align", if C has more than 2 sequences.
15: Otherwise, align read r to each sequence of C. STOP.
16: $p :=$ random(1,$|r|$)
17: **return** "fail to align"

Extending common substrings to align reads to referenced genomes

Using long exact common substrings as seeds to align reads to genomes is similar to [3,12]. Our approach promises to be efficient because instead of exhaustively traversing indices of a read to find optimal common substrings, we find common substrings with respect to random index p of the read.

In Algorithm 2, we iterate at most A times to find long common substrings between $\mathcal{S}$ and each read r. In each iteration, given a random position p, we invoke Algorithm 1 to find the longest common substrings (M) of $\mathcal{S}$ that match to a substring of r with respect to p. As illustrated in Figure 2, each string $s \in M$ corresponds exactly to a substring $r_{i...j}$ of r. This exact match between $r_{i...j}$ and s, naturally, pairs up $r_{1...i-1}$ (a prefix of r) to s_L, the corresponding substring of $\mathcal{S}$ preceding s, and $r_{j+1...m}$ (a suffix of r) to s_R, the corresponding substring of $\mathcal{S}$ following s. If the total edit distances of these two pairs are less than the previously calculated upper-bound t, we align r to the corresponding substring of $\mathcal{S}$.

Note that in the first iteration, the position p is 1 and not a random index of r. The reason for this is that we would like the method of finding long common substrings (Algorithm 1) to be *symmetrical* in the sense that b and f could "wrap around" r. In other words, when $p = 1$, b is a suffix of r and f is a prefix of r. In this case, the concatenation of $b \oplus f$ is not a contiguous substring, but rather two contiguous strings separated by a big gap. This conceptualization of "wrapping around" the read, or thinking of it as a circular instead of linear string, turns out to be quite effective in practice. In many cases, $p = 1$ leads to very long common substrings that lead to correct alignments of reads.

If we cannot align r to any substring of $\mathcal{S}$ after A attempts, then r is unaligned to $\mathcal{S}$, the reference genome. So, it is important to choose A appropriately. If A is too small, there will be many unaligned reads. If A is too large, the algorithm is slow. To select an appropriate value of A, let us again assume that the read and its correct alignment to the genome differ in d places (again $d \leq t$), consequently diving the reads into $d + 1$ blocks. We want to select a value for A so that the longest block (longest common substring) can be sampled with high certainty. The probability that the longest block is selected (i.e. if a random index p lands inside it) is $\frac{m^*}{m}$, where m^* is the length of the longest block. On the other hand, the Pigeonhole Principle dictates that $m^* \geq \frac{m}{d+1}$ (Otherwise, the total lengths of $d + 1$ blocks would be less than m.) This means, $d + 1 \geq \frac{m}{m^*}$, which is the expected number of iterations to sample p to select the longest block.

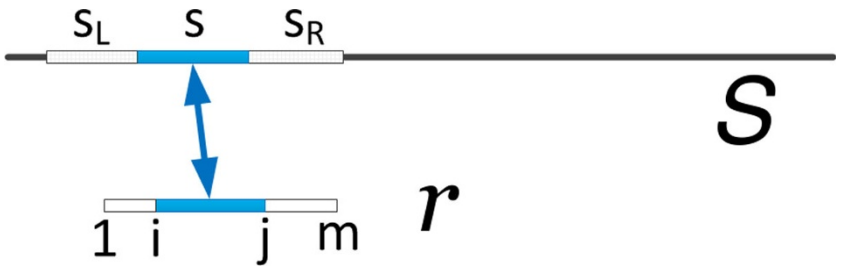

Figure 2 Illustration of Algorithm 2: extending common substring to alignment. Alignment of a read r to the reference genome $\mathcal{S}$ by extending a common substring of r and $\mathcal{S}$ (found in Algorithm 1). There are generally many substrings of $\mathcal{S}$ that match identically to the substring $r_{i...j}$ of r.

Thus, setting $A = t + 1 \geq d + 1$, the longest common substring between a read and the genome is sampled expectedly after A iterations. Further, if $A = c \ldots (t + 1)$, then the probability of landing in the longest block is exponentially increased as a function of c. Trading for speed, $c = 1$ seems to work fine in practice, because even if Algorithm 1 does not return the longest common substring, it is often possible to extend it to find the correct alignment for the read. But longest common substrings minimizes the chance of running into repeats in the genome; i.e. common substrings upon which extensions will lead to incorrect alignments.

Fast heuristic for computing edit distances

Computing edit distances consumes much time of the alignment algorithm (Algorithm 2). In steps 10-11 of Algorithm 2, we compute the edit distance between a read and a substring of the genome and discard it if the distance is greater than t. As each read often match with few substrings of the genome, we expect that such edit distances often exceed t. Examining lines 10-11 of Algorithm 2, we see that actually we do not need to compute the exact value of $d(x, y)$, the edit distance of x and y, as long as we can answer correctly the query $d(x, y) \leq t$.

We claim that the edit distance of x and y, $d(x, y) \leq t$ if and only if Bound$(x, y, t) \leq t$, where Bound is defined in Algorithm 3. To see this, observe that

- If $d(x, y) \leq t$, then Bound(x, y, t) returns $d(x, y)$.
- If $d(x, y) > t$, then Bound(x, y, t) returns either $d(x, y)$ or $t + 1$. The only difference between Bound and the conventional edit distance lies in line 6 of Algorithm 3. Analyzing line 5, we see that once $d_{i,j} > t$ for $1 \leq j \leq m$ (line 6), then $d_{m,m} > t$.

If $d(x, y) > t$, Bound(x, y, t) might not compute the edit distance correctly. Nevertheless, $d(x, y) \leq t$ if and only if Bound$(x, y, t) \leq t$. For aligning reads to bacterial genomes, Bound is much faster than the worst-case complexity $\Theta(m^2)$.

Algorithm 3 Bound(x, y, t)
1: $d_{i,0} := 0$ for $0 \leq i \leq |x|$
2: $d_{0,j} := 0$ for $0 \leq j \leq |y|$
3: **for** $i := 0$ to $|x|$ **do**
4: **for** $j := 1$ to $|y|$ **do**
5: $d_{i,j} := \min(d_{i-1,j-1}+(xi == y_j), d_{i-1,j}+ 1, d_{i,j-1}+1)$
6: **return** $t + 1$ if $d_{i,j} > t$ for $1 \leq j \leq \max\{|x|, |y|\}$
7: **return** $d_{|x|,|y|}$

Results

RandAL is implemented in C++; FM-index codes are adapted from an external library (http://code.google.com/p/fmindex-plus-plus). We compared our method against several aligners including Bowtie [1], BWA [2], Bowtie2 [3], BWA-SW [4], and CUSHAW2 [12]. We chose these methods based on the fact that they are recently published, very popular and their software are available. Comparison tests were conducted on a workstation with two Intel Xeon E5-2680 2.70GHz CPU and 64 GB RAM.

Each aligner is tested with 100,000 simulated reads generated for each of 6 bacterial genomes and 6 chromosomes of eukaryotic genomes. Sizes of these genomes range from 1.3 and 28 millions bases; see Table 1. Genomes were obtained from EMBL-EBI (http://www.ebi.ac.uk/genomes). Since recent sequencing technologies produce read lengths ranging from 35 to 400bp at greater speed and lower cost than previous technologies (e.g. Sanger sequencing) [14], we choose this range of read lengths to evaluate the methods. More specifically, the reads were generated at lengths 35, 51, 76, 100, 200, and 400 as these lengths have been mentioned in the literatures. The *wgsim* C program, part of the SAMtool package [15], was used to generate reads.

Extensive comparisons were performed using SAMtool's default settings, with base error rate at 2%; 15% of polymorphisms are indels with lengths drawn from a geometric distribution with density $0.7 * 0.3^{l-1}$. Additionally, we present summary results for 1% and 4% base error rates with similar trends and conclusions.

Aligned reads from aligners are evaluated using the *wgsim_eval* Perl script, a part of the SAMtool package, using the default setting in which a read is mapped correctly if its mapping position is within a distance of 20 from the correct position. To compare alignment quality, we defined:

$$Precision = \frac{\#\text{ correctly aligned reads}}{\#\text{ aligned reads}}$$

$$Recall = \frac{\#\text{ correctly aligned reads}}{\#\text{ reads}}$$

Alignment quality of 6 aligners

At the outset, we found that Bowtie and BWA were decent performers when read lengths were short, i.e. between 31-56 bases. When read length increased, however, these two aligners were not competitive. Figure 3 shows the average performance (*precision* in the y-axis versus *recall* in the x-axis) of 6 aligners on 6 bacterial genomes and 6 eukaryotic genomes, respectively. Both BWA and Bowtie suffered from a decrease of precision as read length increases. For BWA, recall peaked at around 94% even at longer reads. Bowtie did better at recall for longer reads than BWA, but it was still not competitive to the other 4 aligners, including RandAL. Its bad performance at longer reads is unacceptable because technological trends tend to produce longer reads. For this

Table 1 Reference genomes, obtained from EMBL-EBI (http://www.ebi.ac.uk/genomes).

	Genome	Accession #	Size (bp)
Bacteria	*Wolbachia endosymbiont of Drosophila melanogaster*	AE017196	1,267,782
	Staphylococcus aureus subsp. aureus TW20	FN433596	3,043,210
	Escherichia coli 042	FN554766	5,241,977
	Pseudomonas aeruginosa LESB58	FM209186	6,601,757
	Streptomyces hygroscopicus subsp. jinggangensis 5008	CP003275	10,145,833
	Sorangium cellulosum So ce56	AM746676	13,033,779
Eukaryota	*Debaryomyces hansenii CBS767 chromosome A*	CR382133	1,249,940
	Ectocarpus siliculosus strain Ec 32 chromosome LG01	FN649726	3,745,584
	Schizosaccharomyces pombe chromosome I	CU329670	5,579,133
	Caenorhabditis elegans chromosome I	BX284601	15,072,434
	Taeniopygia guttata chromosome 10	CM000527	20,806,668
	Drosophila melanogaster chromosome 3R	AE14297	27,905,053

reason, we will drop them out of head-to-head comparisons at this point, and will only compare the 4 best aligners: Bowtie2, BWA-SW, CUSHAW2 and RandAL.

A closer look at Figure 3 reveals that BWA-SW was relatively competitive but come roughly in the last place. There is no consistent winner (in terms of both precision and recall) among the top 3 performers, Bowtie2, CUSHAW2, and RandAL. Nevertheless, we can see that RandAL did noticeably better in terms of precision and was still competitive in terms of recall. Importantly, we see that across the wide range of read lengths from 35 to 400 for both bacterial and eukaryotic genomes, the performance of RandAL was consistently high in terms of both precision and recall; average precision was never below 0.98 and average recall was never below 0.95. This consistency distinguishes RandAL from the other top aligners.

An even closer look at individual bacterial and eukaryotic genomes (Figure 4) further reinforces the consistency in performance of RandAL. The lowest precision RandAL got in all 12 genomes was about 0.96, and the lowest recall was about 0.90. In comparison, for the other top performers, the lowest precision was about 0.93 and lowest recall was about 0.80.

All top 4 aligners perform really well in both precision and recall as read length increases. Their performance was quite similar at 400 read length. At shorter read lengths, however, RandAL outperformed the rest, often in both precision and recall.

Rates of misalignment of top 4 aligners

Misalignment means aligning a read at an incorrect position. Misalignment increases the likelihood of running into problems later when we are interested in assembling reads into a complete genome and to identify where the constructed genome different from the reference genome (SNP calling).

The misalignment rate is calculated by dividing the number of incorrect aligned reads by the total number of reads. Figure 5 shows that averaging across all bacterial and eukaryotic genomes, RandAL got noticeably lower misalignments than the other aligners at all different read lengths. This result suggests that RandAL will likely work well with other tools and methods that require alignment of reads to reference genomes.

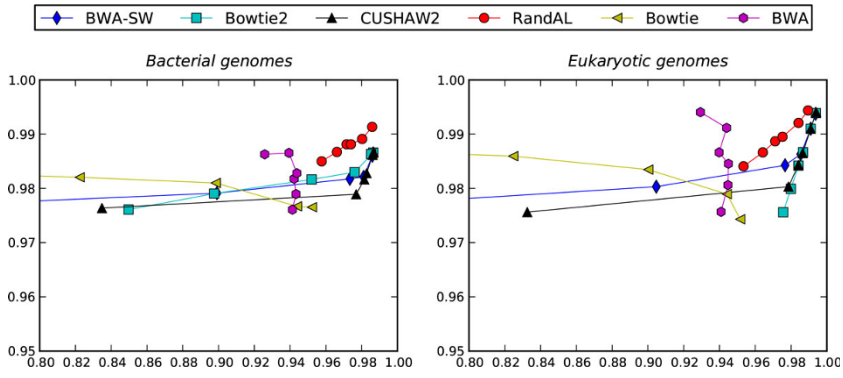

Figure 3 Alignment performance across 6 different read lengths. Recall (x-axis) versus Precision (y-axis) averaged across bacterial genomes and eukaryotic genomes, respectively, at read lengths of 35 bp, 51 bp, 76 bp, 100 bp, 200 bp and 400 bp.

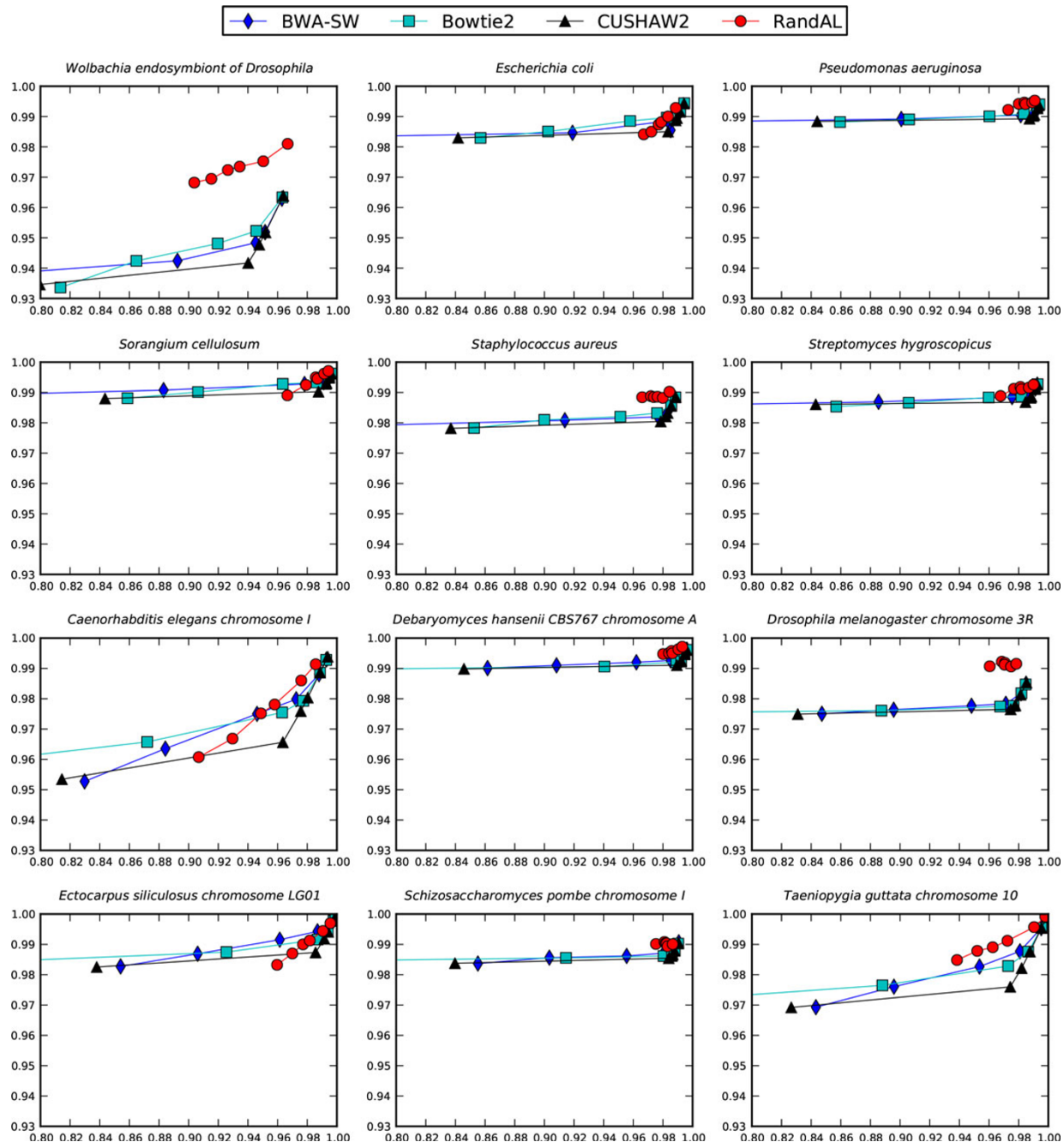

Figure 4 Recall versus precision of top 4 aligners. Performance of top aligners on bacterial genomes (top 6 figures) and eukaryotic genomes (bottom 6 figures). X-axis is recall; Y-axis is precision.

Alignment quality at different base error rates

We have compared performance of 4 different methods using a base error rate of 2%; each nucleotide is mutated with the probability of 2%. Due to the lack of space, we cannot present a comprehensive comparison at different base error rates, as we have at 2%. Nevertheless, analyses at different base error rates show similar behaviors as we have observed at 2% error rates. We present a summary analysis at two other base error rates of 1% and 4% at read lengths of 35 bp, 100 bp, and 400 bp. Table 2 summarizes the average precision and recall of the top 4 aligners. These numbers suggest the followings:

1 All methods performed well at 1% base error rate.
2 With 4% base error rates, the other methods suffered, particularly with shorter reads. The best of

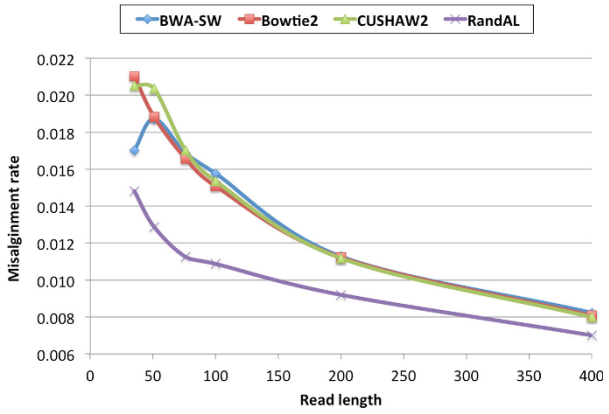

Figure 5 Rate of misalignment. Rate of misalignment averaged across bacterial and eukaryotic genomes.

them (Bowtie2) got ~63% recall at 35 bp. Low recall rate means few reads (out of the total number) were aligned correctly.

3 Our method consistently achieved the highest performance (or among the highest performance) across different read lengths and base error rates. In precision, our method always got the highest, consistently above 97.8%. In recall, even at worst case of 4% base error rate and 35 bp read length, we got ~94%.

Raw running times of top 4 aligners

Theoretically, asymptotic complexity of our method in aligning a read of length m is proportional to $m + m^2$. The worst case complexity of m^2 is due to edit distance computation. The heuristic for computing edit distance, however, reduces this worst-case complexity significantly in practice. Our testing showed that the running times of other methods, like ours, did not depend much on genome sizes.

Table 3 shows the averaged running times (in seconds) of the 4 aligners in aligning 100,000 reads. Our method suffered with shorter reads, but were the second fastest with longer reads (≥ 100 bp). It seems that the benefit of randomization becomes more evident with longer reads.

Table 3 Average running times of top 4 aligners at different read lengths.

	35 bp	51 bp	76 bp	100 bp	200 bp	400 bp
BWA-SW	8.1	13.4	21.6	30.1	56.9	105.2
Bowtie2	2.8	4.1	5.8	8.1	18.3	41.6
CUSHAW2	4.2	7.8	12.7	19.3	67.8	228.5
RandAL	11.1	12.9	13.6	14.5	26.2	81.6

Bowtie2 was the fastest across the board, but as shown in the previous section, its alignment quality is not as good as our method or CUSHAW2. Compared to ours, CUSHAW2 was significantly slower. Observing running times at different read lengths, we speculate that CUSHAW2 might be much be slower than ours with longer reads.

Difficulty of alignment in the presence of repeats

Although eukaryotic genomes are expected to be harder to align than bacterial genomes, an examination of performance of the top 4 aligners in Figure 4 reveals that these aligners did not always perform better on bacterial genomes; eukaryotic genomes were not always harder to align. To quantify the degree of difficulty in aligning reads to genomes, we define *repeat density* as a measure of how many repeats a genome has. Since repeats directly affect alignment quality, the notion of repeat density is meant as a quantifiable approximation of genome complexity. More precisely, given a genome $\mathcal{S}$ and one of its length-k substrings, l, let $f(l)$ be the number of times l occurs in S. We define the *k-mer density* of $\mathcal{S}$ given k to be

$$D(\mathcal{S}|k) = \frac{\sum_{l \in \mathcal{S}, f(l) \geq 2} f(l)}{|\mathcal{S}| - k + 1}$$

$D(\mathcal{S}|k)$ can be interpreted as the probability that a random read of length k is a repeat. The larger $D(\mathcal{S}|k)$ is, the more repeats $\mathcal{S}$ has and the harder it is expected

Table 2 Average precision and recall at 1% and 4% base error rates.

		35 bp		100 bp		400 bp	
		Precision	Recall	Precision	Recall	Precision	Recall
1% base error	BWA-SW	97.60	82.86	98.30	98.29	98.98	98.98
	Bowtie2	97.60	93.40	98.31	98.25	99.00	**99.00**
	CUSHAW2	97.59	92.81	98.33	**98.33**	98.99	98.99
	RandAL	**98.88**	**95.49**	**99.09**	97.04	**99.18**	98.45
4% base error	BWA-SW	97.64	44.93	98.31	97.05	98.97	**98.96**
	Bowtie2	97.61	62.92	98.32	91.62	98.96	98.94
	CUSHAW2	97.67	60.67	98.34	**98.12**	98.95	98.95
	RandAL	**97.80**	**93.55**	**98.66**	97.48	**99.08**	98.48

Table 4 Repeat density of genomes, $D(S|k)$, at various length k.

	Genome	Repeat density at various k					
		35	51	76	100	200	400
Bacteria	*Wolbachia endosymbiont....*	0.181	0.161	0.144	0.134	0.107	0.077
	Staphylococcus aureus...	0.064	0.058	0.053	0.050	0.043	0.036
	Escherichia coli 042	0.053	0.044	0.036	0.031	0.023	0.017
	Pseudomonas aeruginosa ...	0.041	0.037	0.033	0.031	0.026	0.021
	Streptomyces hygroscopicus ...	0.046	0.042	0.038	0.036	0.031	0.025
	Sorangium cellulosum ...	0.038	0.030	0.024	0.020	0.015	0.011
Eukaryota	*Debaryomyces hansenii ...*	0.036	0.032	0.028	0.025	0.019	0.013
	Ectocarpus siliculosus ...	0.092	0.073	0.056	0.046	0.030	0.020
	Schizosaccharomyces pombe ...	0.050	0.047	0.045	0.042	0.036	0.030
	Caenorhabditis elegans ...	0.138	0.105	0.080	0.066	0.039	0.024
	Taeniopygia guttata ...	0.129	0.100	0.070	0.050	0.017	0.002
	Drosophila melanogaster ...	0.068	0.065	0.062	0.060	0.052	0.042

Table 5 Pearson correlation coefficients of repeat density and performance

		k = 35	k = 51	k = 76	k = 100	k = 200	k = 400
Correlation of repeat density and precision	BWA-SW	-0.94	-0.95	-0.96	-0.95	-0.97	-0.95
	Bowtie2	-0.94	-0.95	-0.96	-0.96	-0.97	-0.95
	CUSHAW2	-0.94	-0.94	-0.95	-0.95	-0.95	-0.93
	RandAL	-0.84	-0.83	-0.87	-0.88	-0.93	-0.94
Correlation of repeat density and recall	BWA-SW	-0.64	-0.37	-0.90	-0.95	-0.97	-0.95
	Bowtie2	-0.95	-0.94	-0.96	-0.97	-0.97	-0.96
	CUSHAW2	-0.95	-0.94	-0.95	-0.95	-0.95	-0.93
	RandAL	-0.95	-0.96	-0.97	-0.97	-0.97	-0.96

for aligners to align k-mer reads to $\mathcal{S}$. To investigate how much repeat density correlates with the difficulty of aligning short reads to genomes, first, we computed $D(\mathcal{S}|k)$ for k at each read length 35, 51, 76, 100, 200, and 400. To get a glimpse of its distribution, we show the values of $D(\mathcal{S}|k)$ of the bacterial and eukaryotic genomes, for k's equal to these read lengths, in Table 4. Second, for each k, we computed the Pearson correlation between $D(\mathcal{S}|k)$ of all bacterial and eukaryotic genomes and the performance (precision and recall) of each aligner on aligning reads of length k to the genomes.

Table 5 shows that repeat density is correlated highly negatively to performance (precision and recall) of all aligners. This means the larger $D(\mathcal{S}|k)$ is, the worse any aligner will perform. In other words, $D(\mathcal{S}|k)$ is good indicator of alignment difficulty. That said, we also observe that for BWA-SW at small lengths (k = 35, 51), the negative correlation was weakest. This is probably due to BWA-SW trimming short reads.

Conclusions

We introduced RandAL, a novel randomized approach to aligning reads to reference genomes. We showed that it performed among some of the top aligners that currently exist. Unlike the other aligners, however, RandAL distinctly performs consistently well across a wide range of parameters (read lengths and error rates) across all tested bacterial and eukaryotic genomes. As current sequencing technologies can produce reads in the tested range at low cost [14], our approach promises to work well with these technologies.

Using repeat density as a measure of genome complexity, we showed that this measure correlated highly negatively with alignment quality (precision and recall). This implies that for larger and more complex genomes with many more repeats, these aligners will similarly suffer, as expected.

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

VP conceived and designed the algorithm and experiments. NSV, QT, and NN implemented the algorithm. NSV and QT collected data, implemented experiments, carried out evaluation and data analysis. All authors read and approved the manuscript.

Acknowledgements

This research was partially supported by NSF grant CCF-1320297 to VP.

Declarations

Publication charges for this work were funded by NSF grant CCF-1320297 to VP.

This article has been published as part of *BMC Genomics* Volume 15 Supplement 5, 2014: Selected articles from the Third IEEE International Conference on Computational Advances in Bio and Medical Sciences (ICCABS 2013): Genomics. The full contents of the supplement are available online at http://www.biomedcentral.com/bmcgenomics/supplements/15/S5.

Authors' details
[1]Department of Computer Science, University of Memphis, Memphis, TN 38152, USA. [2]Bioinformatics Program, University of Memphis, Memphis, TN 38152, USA.

Published: 14 July 2014

References

1. Langmead B, Trapnell C, Pop M, Salzberg SL, *et al*: **Ultrafast and memory-efficient alignment of short dna sequences to the human genome.** *Genome Biol* 2009, **10(3)**:25.
2. Li H, Durbin R: **Fast and accurate short read alignment with burrows-wheeler transform.** *Bioinformatics* 2009, **25(14)**:1754-1760.
3. Langmead B, Salzberg SL: **Fast gapped-read alignment with bowtie 2.** *Nature Methods* 2012, **9(4)**:357-359.
4. Li H, Durbin R: **Fast and accurate long-read alignment with burrows-wheeler transform.** *Bioinformatics* 2010, **26(5)**:589-595.
5. Li H, Ruan J, Durbin R: **Mapping short dna sequencing reads and calling variants using mapping quality scores.** *Genome research* 2008, **18(11)**:1851-1858.
6. Li R, Li Y, Kristiansen K, Wang J: **Soap: short oligonucleotide alignment program.** *Bioinformatics* 2008, **24(5)**:713-714.
7. Rumble SM, Lacroute P, Dalca AV, Fiume M, Sidow A, Brudno M: **Shrimp: accurate mapping of short color-space reads.** *PLoS computational biology* 2009, **5(5)**:1000386.
8. Homer N, Merriman B, Nelson SF: **Bfast: an alignment tool for large scale genome resequencing.** *PLoS One* 2009, **4(11)**:7767.
9. Rizk G, Lavenier D: **Gassst: global alignment short sequence search tool.** *Bioinformatics* 2010, **26(20)**:2534-2540.
10. Ahmadi A, Behm A, Honnalli N, Li C, Weng L, Xie X: **Hobbes: optimized gram-based methods for efficient read alignment.** *Nucleic Acids Research* 2012, **40(6)**:41-41.
11. Weese D, Emde AK, Rausch T, Doring A, Reinert K: **Razers–fast read mapping with sensitivity control.** *Genome Research* 2009, **19(9)**:1646-1654.
12. Liu Y, Schmidt B: **Long read alignment based on maximal exact match seeds.** *Bioinformatics* 2012, **28(18)**:318-324.
13. Ferragina P, Manzini G: **Indexing compressed text.** *J ACM* 2005, **52(4)**:552-581.
14. Schatz M, Delcher A, Salzberg S: **Assembly of large genomes using second-generation sequencing.** *Genome Research* 2010, **20(9)**:1165-1173.
15. Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer N, Marth G, Abecasis G, Durbin R, Subgroup GPDP: **The sequence alignment/map format and samtools.** *Bioinformatics* 2009, **25(16)**:2078-2079.

doi:10.1186/1471-2164-15-S5-S2
Cite this article as: Vo *et al.*: RandAL: a randomized approach to aligning DNA sequences to reference genomes. *BMC Genomics* 2014 **15**(Suppl 5):S2.