Bioinformatics, 2016, 1–9
doi: 10.1093/bioinformatics/btw371
Advance Access Publication Date: 4 July 2016
Original paper

OXFORD

Sequence analysis

deBGA: read alignment with de Bruijn graph-based seed and extension

Bo Liu[1,†], Hongzhe Guo[1,†], Michael Brudno[2,3,4] and Yadong Wang[1,*]

[1]Center for Bioinformatics, Harbin Institute of Technology, Harbin, Heilongjiang 150001, China, [2]Department of Computer Science, University of Toronto, ON M5S 3G4, Canada, [3]Genetics and Genome Biology Program, The Hospital for Sick Children, Toronto, ON M5G 1L7, Canada and [4]Centre for Computational Medicine, The Hospital for Sick Children, Toronto, ON M5G 1L7, Canada

*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.
Associate Editor: Inanc Birol

Received on November 30, 2015; revised on May 18, 2016; accepted on June 5, 2016

Abstract

Motivation: As high-throughput sequencing (HTS) technology becomes ubiquitous and the volume of data continues to rise, HTS read alignment is becoming increasingly rate-limiting, which keeps pressing the development of novel read alignment approaches. Moreover, promising novel applications of HTS technology require aligning reads to multiple genomes instead of a single reference; however, it is still not viable for the state-of-the-art aligners to align large numbers of reads to multiple genomes.
Results: We propose de Bruijn Graph-based Aligner (deBGA), an innovative graph-based seed-and-extension algorithm to align HTS reads to a reference genome that is organized and indexed using a de Bruijn graph. With its well-handling of repeats, deBGA is substantially faster than state-of-the-art approaches while maintaining similar or higher sensitivity and accuracy. This makes it particularly well-suited to handle the rapidly growing volumes of sequencing data. Furthermore, it provides a promising solution for aligning reads to multiple genomes and graph-based references in HTS applications.
Availability and Implementation: deBGA is available at: https://github.com/hitbc/deBGA.
Contact: ydwang@hit.edu.cn
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

A fundamental challenge of high-throughput sequencing (HTS) technology is the quick and accurate alignment of sequencing reads to one or more reference genomes (Li and Homer, 2010). Read alignment is one of the most compute-intensive steps of HTS analysis (Langmead and Salzberg, 2012), and is becoming increasingly rate-limiting as HTS technology becomes ubiquitous and the volume of data from each sequencing run continues to rise. Moreover, as more genomes are available (Pruitt *et al.*, 2014), promising new applications of HTS technology require aligning reads to multiple genomes instead of a single reference. For example, the genomes of the typical strains of a species are used as gold standard reference points (Didelot *et al.*, 2012), and species or pathogens can be precisely identified and characterized by simultaneously aligning reads from isolated or metagenomic samples to these reference genomes (Didelot *et al.*, 2012; Fricke and Rasko, 2014; Breitwieser *et al.*, 2015), which has enormous potentials. Similarly, reads from mouse xenograft models typically need to be aligned to both the human and the mouse genome. However, it is still not viable to align large numbers of HTS reads to multiple divergent genomes of various strains or species simultaneously, primarily due to two limitations. Firstly, most state-of-the-art aligners are optimized for a single genome reference. Although they can support almost any reference sequence, appending genomes together to make a longer reference typically degrades performance and decreases accuracy, as many reads have multiple, equally good matches. Secondly, efforts have

been made to support pan-genome indexing (Schneeberger *et al.*, 2009; Huang *et al.*, 2013; Siren *et al.*, 2014; Dilthey *et al.*, 2015); however, most of the proposed methods organize genomes through collinear multiple sequence alignment (MSA). These approaches are useful for highly similar genomes, such as multiple human genomes (1000 Genomes Project Consortium, 2012), but not the genomes of more distant strains or species. Such distant homologues often have significant structural variants (SVs) (Medini *et al.*, 2005; Didelot *et al.*, 2012) (such as inversions, translocations and duplications), and collinear MSA could be not suitable for representing these non-collinear events (Kehr, *et al.*, 2014). In this situation, these approaches may not be applicable to align reads against multiple divergent genomes, e.g. a reference composed by a set of microbial genomes. Moreover, proposed pan-genome indexing methods also have additional limitations. GenomeMapper (Schneeberger *et al.*, 2009) and BWBBLE (Huang *et al.*, 2013) can only integrate relatively small variants (such as SNPs and small indels). The size of generalized compressed suffix array index (Siren *et al.*, 2014) could exponentially increase, which could not be suited to handle the variants in repetitive regions (Valenzuela *et al.*, 2015). Dilthey *et al.* (2015) proposed an augmented MSA-based graph (named as population reference graph, PRG) to better support diverse kinds of variants. In this approach, the de Bruijn graph of the reads is aligned with PRG to infer the donor genome. This approach falls between *de novo* assembly and the alignment of genome sequences. It is suited to the inference of the complex genomic regions, such as the MHC region. However, it is mainly designed for local regions, and due to the lack of effective approach to index the PRG of whole genome, it is still non-trivial to extend this approach to the read alignment against whole genome. Besides the aforementioned limitations, MSA-based aligners also have lower speed, e.g. 10–100 times slower than state-of-the-art aligners (Huang *et al.*, 2013; Siren *et al.*, 2014). Meanwhile, the MSA itself may also affect the result of read alignment, as there is still not a standard method to implement the MSA of genomes (Paten *et al.*, 2014).

Most of the state-of-the-art generic aligners are based on seed-and-extend approach (Li and Durbin, 2010; Dobin *et al.*, 2012; Langmead and Salzberg, 2012; Marco-Sola *et al.* 2012; Mu *et al.*, 2012; Li, 2013). They are superior to backtracking suffix tries [utilized, e.g. BWA (Li and Durbin 2009)] for the increasing read length of HTS technologies, which requires large edit distance. Generally, seed-and-extend aligners consist of two steps: (i) seeding: the inference of putative read positions (PRPs) from the matches (hits) of tokens (seeds) between the read and reference genome; and (ii) extension: alignment of the read to the region surrounding each PRP to determine the most likely read position(s). One of the major bottlenecks faced by this approach is how to handle repetitive genomic regions, even in the context of a single genome, for example, over 50% of the human genome comprised repeats (Treangen and Salzberg, 2012). Because of repeats, many seeds have numerous hits, and the cost of extending all of them is prohibitively high. This problem is even more serious for a reference consisting of multiple genomes, as it will be even more repetitive due to the homologous sequences. Various strategies have been developed to handle hits more effectively, including heuristically filtering the hits (Langmead and Salzberg, 2012; Mu *et al.*, 2012), limiting the number of hits per seed (Li and Durbin, 2010; Marco-Sola *et al.* 2012) or merging hits before extension (Li and Durbin, 2010; Dobin *et al.*, 2012; Li, 2013). However, as all of these strategies process each hit independently, it could be also expensive to evaluate, sort and prioritize the numerous hits. After these steps, however, there could be still a number of candidates left for extension due to the repeats, and the cost of separate alignment of the read to each of them could still be prohibitive.

To improve the alignment of reads to multiple references, as well as the handling of repeats within a single genome, we propose the de Bruijn Graph-based Aligner (deBGA), which uses a graph-based seed-and-extension algorithm to align reads to one or more genomes that have been organized and indexed by a Reference de Bruijn graph (RdBG). de Bruijn graph and A-Bruijn graph were used for the alignment of genomes (Raphael *et al.*, 2004); however, they have not been fully utilized for HTS read alignment. deBGA takes the advantage of the non-branched paths of RdBG (i.e. unipaths; Gnerre *et al.*, 2011) to recognize similar seeds and identical local sequences, which enables it to simultaneously evaluate multiple PRPs located in the same unipath, regardless of the repetitiveness of the corresponding seeds. This greatly reduces the computational burden caused by repeats. Due to its handling of repeats, deBGA is substantially faster than other state-of-the-art aligners. Furthermore, to our knowledge, deBGA is the first viable solution which can align reads to multiple genomes with high throughput and without restriction on the types of variants.

2 Materials and methods

2.1 Overview of the deBGA method

deBGA builds an RdBG to organize one or more reference genomes, and uses a hash table-based data structure, Reference de Bruijn graph Index (RdBG-Index), to index the unipaths of the RdBG instead of the original sequence (Fig. 1a and Supplementary Fig. S1). This novel genome index provides two fundamental operations to

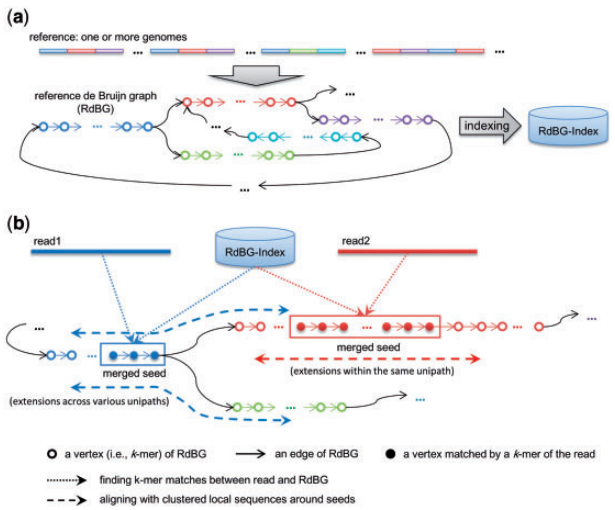

Fig. 1. A schematic illustration of the deBGA method. (**a**) The organization and indexing of reference. The reference (consisting of one or more genomes) is collapsed into an RdBG. As the copies of a *k*-mer within the reference collapse to the same vertex of the RdBG, the repetitive sequences of the reference (the segments with various colors) collapse to various unipaths (the hollow dots with the same colors corresponding to the repetitive sequences). Further, the unipaths are indexed by RdBG-Index, a hash table-based data structure (Supplementary Fig. S1). (**b**) The read alignment method. The *k*-mers of the reads are used as seeds and matched to the vertices of the RdBG through RdBG-Index (the solid vertices). deBGA takes advantage of the property of RdBG to merge seeds having similar PRPs through unipaths, which can efficiently process repetitive seeds (Section 2, Supplementary Fig. S2). For each of the merged seeds (the blue and red rectangles), if all the local sequences around its PRPs are within the same unipath (the case of 'read2'), the read is directly aligned with the unipath (the red dashed arrows) for simultaneously performing the corresponding extensions (Supplementary Fig. S3); otherwise, deBGA clusters the local sequences extending outside the unipath on-the-fly, and separately aligns the read with the clustered local sequences (the blue dashed arrows) (Color version of this figure is available at *Bioinformatics* online.)

recognize and merge similar seeds as well as identical local sequences by taking the advantage of the unipaths of de Bruijn graph. With the RdBG-Index, multiple similar seeds and identical extensions within the same unipath can be simultaneously handled regardless of their repetitiveness. deBGA utilizes the functions of RdBG-Index for implementing a graph-based seed-and-extension approach which can well-handle the repeats within the reference to reduce the cost of read alignment. A schematic illustration of the deBGA method is in Figure 1b.

2.2 Indexing reference genome with RdBG-Index

deBGA initially builds a de Buijn graph of the reference genome (RdBG) with a user-defined k-mer length (*-k* option in the software). The vertices and edges of the RdBG are simply derived from all the $(k+1)$-mers of reference genome. deBGA investigates the in- and out-degrees of the vertices to recognize the starting vertices of the non-branched paths (i.e. the unipaths) of RdBG, and derives the unipaths by traversing the graph from the starting vertices. The sequences of the unipaths (unitigs) are then generated by collapsing all the vertices of the corresponding unipaths. As a unitig may have multiple copies in the reference genome, for each of the unitigs, deBGA also records the starting positions of all its copies. RdBG-Index is then built for indexing the unitigs. It consists of three major data structures (Supplementary Fig. S1b):

i. A linear table (UT_{seq}) records the unitigs, and a specific identifier (U_{ID}) is assigned to each of them.
ii. A hash table ($UT_{k\text{-mer}}$) indexes all the k-mers of the unitigs. For each k-mer, $UT_{k\text{-mer}}$ records its token as well as its unipath coordinate, i.e. the U_{ID} of the unitig which the k-mer occurs, and the offset of the k-mer on the unitig.
iii. A linear table (UT_{pos}) records the starting positions for all the copies of the unitigs.

This index provides three major functions: (i) retrieving the unipath coordinate for a given k-mer (including the offset of the k-mer within the unipath), (ii) retrieving all the genomic positions corresponding to a given unipath coordinate and (iii) retrieving the corresponding unitig for a given unipath. It is worthnoting that, for any k-mer of the reference genome, its unipath coordinate is unique, as it appears exactly once in RdBG. Moreover, the genomic positions of any given k-mer can be obtained by jointly retrieving $UT_{k\text{-mer}}$ and UT_{pos}. Using the three functions, the following two operations are implemented to merge similar seeds as well as reduce identical extensions.

2.2.1 Recognizing similar seeds through unipath decomposition

All the copies of a k-mer within the reference genome collapse to the same vertex of the RdBG. This property indicates that a k-bp-long seed's hit on the reference genome essentially connects the seed and a certain vertex of RdBG, i.e. the hits of the seed are exactly the short token matches on the various copies of the same unitig. Thus, the hit at the ith copy of the unitig, G^i_{hit}, can be represented as $G^i_{\text{hit}} = G^i_U + (O_{\text{hit}} - 1)$, $i = 1, \ldots, C_U$, where C_U is the number of the copies of the unipath, G^i_U is the starting position of the ith copy of the unitig, and O_{hit} is the unitig offset of the seed. Considering that, for each of the hits, a specific PRP can be estimated as a genomic position around $G^i_{\text{hit}} - (O_{\text{read}} - E_{\text{indel}} - 1)$, where O_{read} is the seed's offset on the read and E_{indel} is the effect of indels, all the PRPs can be decomposed by the G^i_U, O_{hit}, O_{read} and E_{indel} values (Supplementary Fig. S2a). It is also worthnoting that when inferring PRPs, most aligners consider the effect of indels (E_{indel}) as a constant (e.g. considering it as a limited size of indels).

In this situation, if two seeds hit the same unipath, it is possible that they have two similar sets of PRPs (Supplementary Fig. S2b). More precisely, for each copy of the unipath, the distance between the corresponding PRPs of the two seeds can be represented as $D^{12}_{\text{PRP}} = \left|(O^2_{\text{hit}} - O^1_{\text{hit}}) - (O^2_{\text{read}} - O^1_{\text{read}})\right|$, where O^1_{read} and O^2_{read} are the read offsets, and O^1_{hit} and O^2_{hit} are the unitig offsets of the two seeds, respectively. This distance is regardless of the G^i_U values, i.e. on all the copies, the distance between the corresponding PRPs is exactly the same. Thus, it is easy to directly assess the difference between the two PRP sets by D^{12}_{PRP}, without separately investigating the specific PRPs. Furthermore, if this value is smaller than the maximal allowed length of indels (usually a few tens of base pairs), the two seeds can be considered as having similar effects to the seed-and-extension process and they can be safely merged without loss of important PRPs.

In practice, it is easy to check whether two seeds connect to the same unipath through $UT_{k\text{-mer}}$. If so, the D^{12}_{PRP} value can be calculated to investigate the similarity of the seeds. deBGA takes advantage of this property to generate and merge seeds without separately investigating their hits.

2.2.2 Recognizing identical local reference sequences through unipath embedding

For a given seed, it is easy to directly investigate the distance between its unipath offset and the two ends of the unipath to determine whether the unitig can accommodate the local sequences to be aligned with (i.e. embedded in the unipath). If so, all the extensions for the seed involve aligning the read to the various copies of the unitig. In such cases, the extensions can be reduced to a specific alignment between the read and the unitig itself, regardless of the number of the hits (Supplementary Fig. S3). In practice, given a seed, deBGA retrieves $UT_{k\text{-mer}}$ and UT_{seq} to determine whether it implies a set of identical local sequences. If it does, deBGA aligns the read against the retrieved unitig to simultaneously accomplish the extensions.

2.3 Workflow of read alignment

A flowchart of read alignment is in Supplementary Figure S4. deBGA aligns a read in several iterations. In each iteration, deBGA generates a set of seeds (i.e. 're-seeding') to adapt to the divergence of the read. The main idea of this iterative process is to use fewer seeds to more efficiently align less divergent reads and to use more seeds in later iterations to sensitively align more divergent reads. Three user-defined parameters, R_S (the number of iterations of re-seeding, *-s* option in the software), I_S (the minimal interval of seeding, *-i* option in the software) and M_{hit} (the maximum allowed number of hits per seed, *-n* option in the software) decide the number of iterations and the generation of seeds in each iteration.

More precisely, in the rth iteration ($r = 1, \ldots, R_S$), deBGA tries to generate seeds every $(R_S - r + 1) \times I_S$ bp on the read. The alignment is implemented in the graph-based seed-and-extension strategy which mainly consists of four steps: (i) using k-mers to find the maximal exact matches between the read and the unipaths of RdBG as seeds, (ii) merging seeds within the same unipath to reduce seeds implying similar PRP sets, (iii) splitting and regrouping the remaining seeds to further merge hits at similar genomic positions and (iv) for each of the regrouped seeds, if all the local sequences around its PRPs are within the same unipath, directly aligning the read against the unipath to simultaneously accomplish the corresponding extensions, otherwise, clustering the local sequences from unipaths on-the-fly, and separately aligning the read against the clustered local

sequences. Steps (i), (ii) and (iv) particularly take advantages of the two operations mentioned above.

In each iteration, if the edit distance of the best end-to-end alignment found is smaller than a pre-defined threshold, C_{ED} (*-c* option in the software), deBGA considers that the read is confidently aligned and stops the process early. The obtained optimal and suboptimal alignments will be outputted as results. However, if the read still cannot be successfully aligned after R_S iterations, deBGA resets the seed density to $R_S \times I_S$ bp, and ignores the limitation on the number of hits per seed, i.e. sets $M_{hit} = \infty$, to deal with very repetitive reads. In practice, for most reads, the algorithm finishes in only a few iterations by finding an alignment with edit distance much smaller than C_{ED}.

It is also worthnoting that there is a heuristic in deBGA for handling very divergent reads. If there is at least one match between seeds and reference in a given iteration, but no successful alignment is obtained, deBGA can dynamically adjust the early-stop condition, depending on the user setting. That is, in the following iterations, deBGA can dynamically adjust C_{ED} according to the user setting of *–cl* option, or accept the best obtained local alignment as a result if the *–local* option is set. If neither of the two options are set (the default setting of deBGA), the heuristic will not be used. This heuristic is beneficial for effectively handling large number of sequencing errors, e.g. simultaneously improving the sensitivity and throughput of the alignment of the reads having many low quality bases.

2.4 Seed generation

deBGA finds maximal exact matches between the read and the unitigs as seeds (called 'Uni-MEM seeds'). A Uni-MEM seed is generated initially by a *k*-mer match from the read to a specific unitig, and the match is subsequently extended until it meets a mismatch or the end of either the read or the unitig (Supplementary Fig. S5a). In practice, for the *r*th iteration ($r = 1, \ldots, R_S$), deBGA extracts the *k*-mers starting at every $(R_S - r + 1) \times I_S$ bp from the read as candidates, and tests each of them from upstream to downstream. As a Uni-MEM may cover multiple candidate *k*-mers, deBGA will neglect the untested positions which have already been fully covered by an existing Uni-MEM.

A Uni-MEM seed can be formulated by a tuple $(S_R, S_{U_I D}, S_{U_{Offset}}, S_{cover}, S_{hit})$. The combination of S_R, $S_{U_I D}$ and $S_{U_{Offset}}$ depicts the connection between the read and the RdBG (called a 'read-RdBG connection'), where S_R indicates the seed's offset on the read, $S_{U_I D}$ and $S_{U_{Offset}}$, respectively indicate the identifier and the offset of the unitig. S_{cover} is a bit-vector recording the read positions covered by the seed, and S_{hit} is the set containing all the hits of the seed.

A Uni-MEM seed can be also seen as an implicit combination of a series of continuous *k* bp seeds (Supplementary Fig. S5b). Because of the property of the RdBG, all such seeds imply identical PRP sets, i.e. they are practically equivalent for the read alignment process, and a Uni-MEM seed implicitly merges them.

2.5 Seed reduction within same unipath

deBGA bins the Uni-MEM seeds in terms of their unipaths to reduce seeds implying similar sets of PRPs. For each bin, deBGA chooses the seed having largest coverage length as the 'merging seed', and checks whether each of other seeds in the bin can be merged. If it can be merged, deBGA removes the seed and merges its S_{cover} vector into that of the merging seed (Supplementary Fig. S6). The merging condition is as following, assuming the read-RdBG connections of the merging seed and the testing seed are, respectively, $\left(S^M_R, S^M_{U_I D}, S^M_{U_{Offset}}\right)$ and $\left(S^T_R, S^T_{U_I D}, S^T_{U_{Offset}}\right)$, if the value of $\left|\left(S^M_{U_{Offset}} - S^T_{U_{Offset}}\right) - \left(S^M_R - S^T_R\right)\right|$ is smaller than the maximal allowed length of indels, the two seeds are considered similar and merged. Here, the maximal allowed length of indels is used for considering the effect of small indels during the merging process, and the parameter is empirically set as 32. When a merging seed cannot merge any remaining seeds in the same bin, deBGA sets it aside and chooses a new one from the remaining seeds to continue this procedure, until no seeds remain.

2.6 Seed merging across unipaths

Seeds within various unipaths may have hits at nearby genomic positions. deBGA merges these similar hits using a split-and-merging approach. For each seed remaining from the last step, deBGA splits it into a set of individual 'hit-seeds', i.e. each of them inheriting the read-RdBG connection and read coverage from the original seed, but only associated with a specific hit. All the hit-seeds are binned in terms of their positions. For each bin, the hit-seed with largest read coverage length is selected, and other hit-seeds are merged into it by simply merging their S_{cover} bit vectors. Next, deBGA regroups the merged seeds which have identical read-RdBG connections and read coverage vectors. The seeds generated by the regrouping are considered as candidate seeds for extension.

To avoid unnecessary extensions, deBGA pre-filters out all candidate seeds with coverage length $\geq$k bp shorter than that of the largest seed. For paired-end reads, deBGA also filters out the hits on each end of the read that cannot be paired with any hit on the opposite end.

2.7 Seed extension with identity checking

For each candidate seed, deBGA investigates the local structure of the RdBG around the seed to recognize whether the corresponding unipath can accommodate the local sequences for extension. If so, a simultaneous extension will occur because the corresponding local sequences are identical (Supplementary Fig. S3). Otherwise, deBGA extracts and clusters all the local sequences around the PRPs of the seed, and separately aligns the read against each of the clustered sequences. It is worthnoting that the employed local sequences are not always as long as the read shown in Supplementary Figure S3. If that happens, deBGA, respectively, extends both sides of local sequences at an extra 32 bp (i.e. the maximal allowed length of indels) to cope with potential indels. Landau–Vishkin (Landau and Vishkin, 1986) and banded Smith–Waterman (Li, 2013) algorithms are used to implement end-to-end and local alignment, respectively. After the extension process, the primary and alternative alignments of the read are chosen based on the alignment score, and a BWA-like mapping quality is also calculated based on the chosen alignments.

2.8 Anchoring alignment for paired-end reads

For paired-end reads, deBGA performs additional processing if the read is unaligned after all the iterations. In such cases, deBGA separately aligns the two ends of the read with the seeds generated in the R_S-th iteration (the seeds are recorded in advance). The optimal and suboptimal alignments of each end are used as anchors, and the opposite read ends are aligned to the local regions around the anchors to compose a paired-end alignment. However, if too many bases of the opposite end cannot be matched in the anchoring alignment, the single-end alignments of the two ends are separately outputted as the result.

3 Results

We benchmarked deBGA with two categories of tasks, i.e. read alignment against multiple genomes and a single genome, to evaluate its ability on various applications. In each of the tasks, a series of simulated and real datasets were employed to assess its speed, sensitivity and accuracy. Six state-of-the-art aligners, Bowtie2, BWA, BWA-MEM, STAR, SeqAlto and GEM, were compared. All the aligners were run with a variety of configurations (Supplementary Notes). The benchmarks were conducted on a server with an Intel Xeon E4820 CPU at 2.00 GHz and 1 TB RAM, running Linux Ubuntu 14.04.

3.1 Read alignment against multiple genomes

3.1.1 Read alignment against multiple genomes of the same species

For many species, there have already been multiple genomes of their various strains assembled. Previous studies (Schneeberger *et al.*, 2009) indicate that it is helpful to align reads against the multiple genomes of the same species to perform precise re-sequencing analysis, especially when the strains are divergent. Moreover, this task is also a fundamental step to read alignment against a library of the genomes of various species (e.g. the RefSeq database), as such libraries usually record more than one genomes for the various strains of the same species.

We assessed the ability of the aligners on such tasks with a series of simulated datasets from *Escherichia coli* strains at first. We downloaded the genomes of all the 62 *E. coli* strains available from NCBI RefSeq database (Pruitt *et al.*, 2014) (http://www.ncbi.nlm.nih.gov/refseq/). Five reference genome libraries, consisting of 1, 10, 20, 40 and 62 strains, respectively, were built (Supplementary Table S2). For each library, Mason Simulator (Döring *et al.*, 2008) was used to simulate 1 million 100 bp Illumina-like paired-end reads from all the involved genomes (insert size: 500 ± 25 bp). Each of the reads could originate from any strain in the library, and was aligned to the corresponding composite reference. The sensitivity, accuracy and speed of the alignment were investigated. The sensitivity is assessed by the proportion of aligned reads; the accuracy is assessed with two metrics, i.e. the proportions of the reads correctly aligned by all the alignments and by the primary alignments; and the speed is assessed using the read alignment time, i.e. the wall clock time of the execution of the aligner deducting the time of loading index, as the cost of loading index is constant (Supplementary Notes).

The results (Fig. 2a and Supplementary Table S3) demonstrate that:

i. deBGA is efficient, especially when the reference is more repetitive. To investigate the repetitiveness of the references, we assessed the number of distinct *k*-mers and the copies per *k*-mer (termed as $\bar{C}_k$) of the references (Supplementary Table S4). It is observed that the $\bar{C}_k$ values obviously increase with more strains involved, indicating that the references with more strains are more repetitive and more difficult to cope with, since a seed would on average have more hits to handle. We also found that the references with ≥ 10 *E. coli* strains have even higher $\bar{C}_k$ values than that of human reference genome (GRCh37/hg19), which is commonly considered as highly repetitive. The results suggest that deBGA has a substantial improvement on alignment speed. This is especially noticeable for datasets with more strains, e.g. on the dataset with 62 strains, it is 2.8–113 times faster than other aligners. This advantage stems from that similar seeds and identical extensions are largely merge-and-reduced with the graph-based seed-and-extension strategy.

ii. Overall, deBGA correctly aligns more reads (98.77–99.95%) than other aligners. For all the aligners, the accuracies of primary alignments dropped with increasing numbers of strains. This is not surprising, as the more strains are involved, the more homologous sequences there are, and the more reads that cannot be confidently aligned. In this situation, aligners that output multiple equally best alignments, like deBGA and BWA (configured to output up to 1000 results per read), will be less affected and perform better (especially on the 62 strains dataset). Bowtie2, SeqAlto, BWA-MEM and STAR correctly align fewer reads, although this is likely due to their inherent design. That is, when reads can be aligned to multiple equally good positions, candidate positions are randomly selected based on the various heuristics adopted by the aligners as a result. In this situation, a proportion of reads could be aligned to wrong strains due to homologous sequences, or wrong positions in the same strain due to the repeats of the same genome.

Two simulated datasets from *Arabidopsis thaliana* strains were employed to further benchmark the aligners on the multiple genomes of a eukaryote organism. One is from the reference genome of *A. thaliana* (TAIR10), and the other one is from TAIR10 plus other 18 *A. thaliana* strains assembled by 1001 Genomes Project (Gan *et al.*, 2011) (the genomes were downloaded from: http://mus.well.ox.ac.uk/19genomes). Similar results are obtained (Fig. 2a and Supplementary Table S5), i.e. deBGA is at least three times faster on both of the one and the 19 genome(s) datasets; meanwhile, it correctly aligned highest number of reads.

We also implemented GenomeMapper (Schneeberger *et al.*, 2009), a read alignment tool specifically designed for multiple genomes, on the simulated datasets from the 62 *E. coli* strains and 19 *A. thaliana* strains. However, we found that although its accuracy is comparable (marginally lower) to state-of-the-art aligners, the tool has a much slower speed (line 161 of Supplementary Table S3 and line 65 of Supplementary Table S5). GenomeMapper costs about 160-folds more alignment time than deBGA to process only a half of the *E. coli* dataset, and nearly 600-folds more time to process one-tenth of the *A. thaliana* dataset. This speed could not be well-suited to handle large amount of HTS data.

To further investigate the effect of repetitiveness on the performance, we benchmarked the aligners with two additional simulated datasets which, respectively, are from two divergent *E. coli* strains [K-12 MG1655 substrain (NC_000913) and O157:H7 strain (NC_002695)] and two highly similar strains [K-12 MG1655 substrain (NC_000913) and K-12 W3110 substrain (NC_007779)]. It is observed that (Supplementary Fig. S7 and Supplementary Table S3) although aligners outputting multiple best positions (e.g. deBGA) can still achieve high overall accuracies on both of the two datasets, all the aligners achieved low accuracies of primary alignments on the dataset from the two similar strains. This is mainly due to that the two similar strains already shared many homologous sequences, so that many reads are hard to be confidently aligned to only one best position. Such results suggest that the accuracy of read alignment against multiple genomes greatly depends on the similarity of the genomes. If the genomes share many common sequences, it would be hard to confidently align the reads; otherwise, the reads are easier to accurately align with confidence.

We used three real datasets from *E. coli* strains to evaluate the performance of deBGA at aligning real reads. Two of them are from *E. coli* K-12 MG1655 substrain (SRA accession: ERR008613 and SRR522163), and one is from *E. coli* O157:H7 strain (SRA accession: SRR530851). For each of the three datasets, 1 million paired-

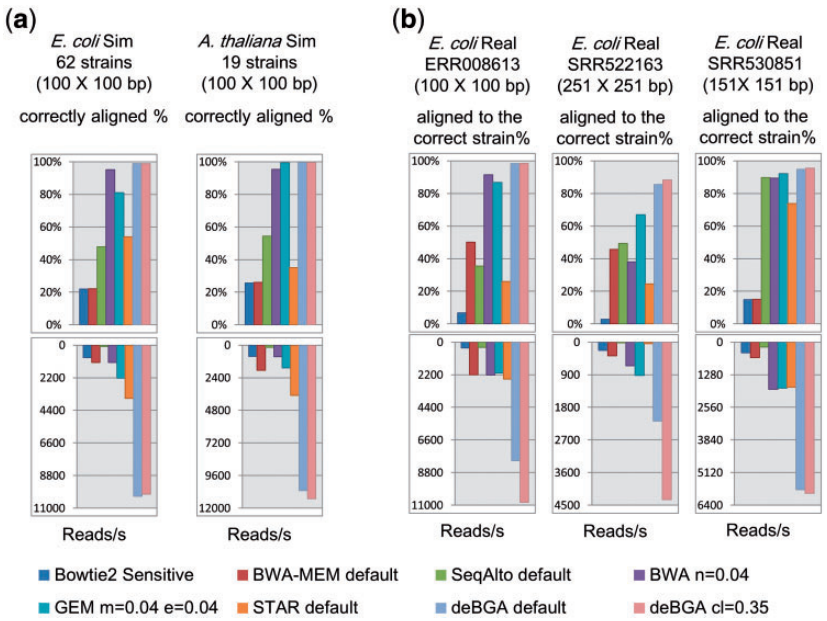

Fig. 2. Benchmarking on the datasets from *Escherichia coli* strains. (**a**) Results on the simulated datasets from 62 *E. coli* strains and 19 *Arabidopsis thaliana* strains; (**b**) results on the three real *E. coli* datasets. For each of the datasets, 'Reads/s' indicates the number of reads aligned per second. 'Correct aligned%' indicates the proportion of reads aligned to their correct positions by at least one alignment. 'Aligned to the correct strain%' indicates the proportion of reads aligned to the correct strain by at least one alignment. Each bar in the figure also corresponds to a line of Supplementary Tables S3, S5 and S6 (the corresponding lines are highlighted)

end reads were randomly extracted and aligned to the reference consisting of all the 62 *E. coli* strains. Two sensitivity metrics were assessed, i.e. one is the proportion of aligned reads, and the other is the proportion of reads aligned to the correct strain. The latter metric is more critical to the read alignment against multiple genomes, and also reflects the accuracies of the aligners. The speeds of the aligners were also assessed by the alignment time.

The results suggest that, with its default setting, deBGA is overall at least three times faster than BWA and GEM, and many times faster than other aligners (Fig. 2b and Supplementary Table S6). Further, a portion of real reads had many low-quality bases, especially the longer Illumina MiSeq datasets (i.e. SRR522163 and SRR530851). deBGA addresses these reads more efficiently using an early-stop heuristic (*–cl* parameter, optimum setting: 0.35, refer to Section 2). With the heuristic, it is about 4–114 times faster than other aligners.

deBGA also aligned more reads to the correct strain (98.55%, 88.38% and 96.39% for the three datasets, respectively) than other aligners, suggesting that it is sensitive. Meanwhile, for all the datasets, the numbers of reads aligned to the correct strains (i.e. NC_000913 for ERR008613 and SRR522163, and NC_002695 for SRR530851) by deBGA are much higher than other distant strains (Supplementary Fig. S8 and Supplementary Table S7). This signal may be useful for a wide variety of applications, such as pathogen identification. Meanwhile, other close homologues such as NC_007779, NC_010473, NC_012759, NC_017625 and NC_017638 for the K-12 MG1655 substrain, and NC_017906, NC_011353, NC_002655 and NC_013008 for the O157:H7 strain, have similar numbers of reads aligned. This is also reasonable as these genomes share many homologous sequences, thus many reads could also have equally best alignments on these genomes.

3.1.2 Read alignment against multiple genomes of various species

There are also many applications requiring aligning reads against the genomes of multiple species. For example, in many metagenomics studies, it needs to align reads to a large set of known genomes; it is also essential to align reads from mouse xenograft models to both of the human and the mouse genomes. Such tasks are still non-trivial to state-of-the-art approaches. We used simulated and real datasets to benchmark deBGA on such tasks, the aligners shown good performance in the previous tasks, i.e. BWA-MEM, BWA, GEM and STAR, are compared.

We used Mason to simulate two datasets for evaluating the ability of the aligners to handle metagenomics reads and xenograft model reads, respectively. One is from all the RefSeq bacteria genomes, and the other one is from the human (GRCh37/hg19) and mouse (MM10) reference genomes. Each has 1 million 100 bp Illumina-like paired-end reads (insert size: 500 ± 25 bp).

Three issues are observed from the results (Fig. 3 and Supplementary Tables S8 and S9), which are quite similar to that of the previous tasks. Firstly, deBGA is at least 4-folds faster on the metagenomics dataset, and obviously faster on the xenograft dataset as well. Considering the $\bar{C}_k$ values of the references (higher than that of human reference genome, Supplementary Table S4), the speed suggests the superior ability of deBGA to handle repeats. Secondly, the sensitivities of the aligners are quite high only except for STAR on the metagenomics dataset, that about 10% of the reads are unaligned. Thirdly, deBGA and GEM achieved equally high overall accuracies, outperforming other competitors. This is also likely due to the different designs of the aligners for repetitive reads.

As there is lack of ground truth for real metagenomics datasets, we built a 'pseudo' real metagenomics dataset by six real sequencing datasets from various bacteria genomes, to assess the ability of the aligners on real data. Each dataset is independently sequenced from a specific bacteria strain recorded in RefSeq (line 15 of Supplement Table S1). A total of 250K paired-end reads were randomly extracted from each of the datasets, i.e. the whole datasets consist of 1.5 million reads. For this dataset, we assessed the speed and the sensitivity. The results (Supplementary Table S10) suggest that,

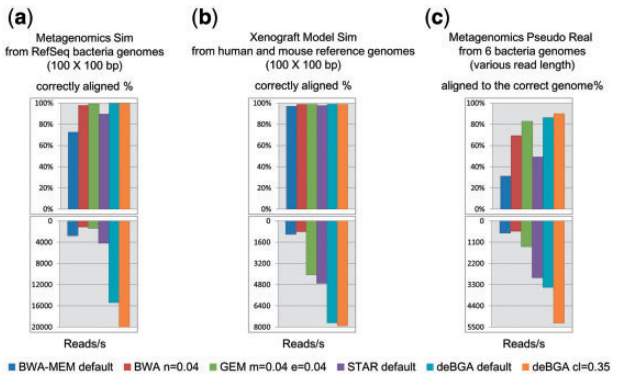

Fig. 3. Benchmarking on metagenomics and xenograft model datasets. (**a**) Results on the simulated metagenomics dataset from RefSeq bacteria genomes; (**b**) results on the simulated xenograft model dataset; (**c**) results on the 'pseudo-real' metagenomics dataset which is composed by six bacteria genome sequencing datasets. For each of the datasets, 'Reads/s' indicates the number of reads aligned per second. 'Correct aligned%' indicates the proportion of reads which are aligned to their correct positions by at least one alignment. 'Aligned to the correct genome%' indicates the proportion of reads which are aligned to the correct bacteria genome by at least one alignment. Each bar in the figure also corresponds to a line of Supplementary Tables S8–S10 (the corresponding lines are highlighted)

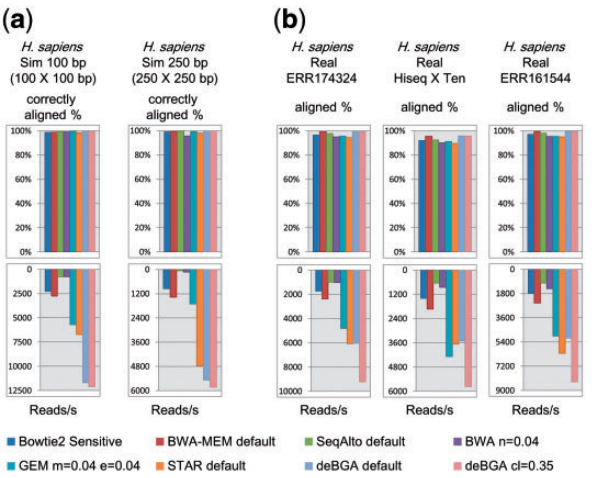

Fig. 4. Benchmarking on the datasets from human genome. (**a**) Results on the 100 and 250 bp simulated human datasets (respectively marked by '*H. sapiens* Sim 100 bp' and '*H. sapiens* Sim 250 bp'); (**b**) results on the three real human datasets (respectively marked by 'ERR174324', 'Hiseq X Ten' and 'ERR161544'). For each of the datasets, 'Reads/s' indicates the number of reads aligned per second. 'Correctly aligned%' indicates the proportion of reads which are aligned to their correct positions by at least one alignment. 'Aligned%' indicates the proportion of reads which are aligned by at least one alignment. Each bar in the figure also corresponds to a line of Supplementary Tables S11 and S14 (the corresponding lines are highlighted). Also refer to Supplementary Figure S9 for a 'zoomed-in' view of the results which more clearly presents the difference between the aligners

Table 1. Statistics on the called SNPs and indels of NA12878

Aligner	Metric	SNP		Indel	
		Raw	VQSR pass	Raw	VQSR pass
deBGA	Total#	4 764 671	3 896 441	447 951	375 932
	GIAB#	2 825 885	2 752 257	87 573	84 954
	GIAB%	99.94%	97.33%	99.32%	96.35%
	dbSNP#	3 814 832	3 258 223	202 781	174 971
BWA-MEM	Total#	4 434 235	3 633 112	438 990	292 648
	GIAB#	2 825 172	2 688 519	87 645	80 413
	GIAB%	99.91%	95.08%	99.40%	91.20%
	dbSNP#	3 648 836	3 124 841	198 368	145 400

Note: 'Raw' and 'VQSR pass', respectively, indicate the variants initially output by GATK HaplotypeCaller and filtered by VQSR. 'GIAB#' and 'GIAB%', respectively, indicate the number and the ratio of the GIAB ground truth variants which are recovered by the called variants. 'DbSNP#' indicates the number of the called variants which can be matched by dbSNP ver. 146.

similar to that of the real *E. coli* reads, deBGA correctly aligned highest number of the 'pseudo' metagenomics reads with fastest speed, indicating that the method is potentially very effective to handle the reads from metagenomics samples. It is also worthnoting that the speed of STAR is comparable; however, it unaligned nearly 50% of the reads and the number of the reads aligned to the correct genomes is even lower, indicating that it may be less sensitive to process metagenomics datasets. This is likely due to the heuristics adopted by STAR which may discard many repetitive reads.

3.2 Read alignment against a single genome

3.2.1 Read alignment against human reference genome

Five simulated and three datasets from human genome (Supplementary Table S1) were aligned to human reference genome (GRCh37/hg19). The five simulated datasets are in various read lengths (100, 125, 150, 200 and 250 bp), which are simulated by Mason based on GRCh37/hg19, to mimic the datasets from extant Illumina platforms. Each has 1 million paired-end reads (insert size: 500 ± 25 bp). Two of the three real human datasets are from the CEU HapMap individual sample NA12878 which are, respectively, sequenced by Illumina HiSeq 2000 (SRA accession number: ERR174324) and HiSeq X Ten platforms (downloaded from Illumina BaseSpace website); and one is from the Yanhuang (YH) sample (Wang *et al.*, 2008) sequenced by Illumina HiSeq 2000 platform (SRA accession number: ERR161544). For each of the three datasets, one million paired-end reads were randomly extracted for benchmarking, and only the sensitivities were assessed by the proportions of aligned reads, due to the absence of a ground truth.

Three conclusions (Fig. 4, Supplementary Fig. S9 and Supplementary Tables S11–S14) can be drawn.

(i) deBGA is overall faster than STAR, especially with the fast $cl = 0.35$ setting, but they are comparable. Compared with other aligners, the superiority of deBGA is more obvious. For example, it is on average as twice as fast as GEM with the default setting $m = 4\%$ (the third fastest tool), even though GEM is less sensitive with this setting. When GEM is configured to be as sensitive as deBGA (e.g. $m = 8\%$), it is 7–13 times slower than deBGA.

(ii) deBGA aligns simulated reads with high accuracy. Only considering the primary alignments, the accuracy of deBGA is comparable to (i.e. only several tenths of percent lower) that of BWA-MEM, which is the most accurate. Further, of those reads not correctly aligned by the primary alignments of deBGA, about 80–90% (on various datasets) were correctly aligned by alternative alignments with the same or even larger edit distances (Supplementary Table S12), indicating that it is hard to make the correct choice for these reads. When considering all the alignments, deBGA aligned almost all (99.67–99.90%) of the reads to their correct positions, more than all other aligners on all but one dataset. On the 100 bp dataset, GEM with $m = 6\%$ and 8% configurations

achieved the same accuracy as deBGA; however, it was 2 and 3.5 times slower, respectively.

As the quality of the alignment of repetitive reads is especially critical, we investigated the accuracy of the alignment of the 100 and 250 bp reads around retrotransposon regions. The results (Supplementary Table S13) indicate that deBGA achieved equally best accuracy for aligning these repetitive reads. This could be beneficial for the analysis of repetitive elements, which is still a difficult task in genomics studies.

(iii) BWA-MEM and deBGA achieved similar sensitivity on real data, outperforming other aligners by 1.5–6.1% (Supplementary Table S14). It is also observed that, with its default settings, BWA-MEM, respectively, clipped over 50% of their bases for 17 814 reads from the NA12878 HiSeq 2000 dataset, 25 407 reads from the NA12878 HiSeq X Ten dataset and 20 381 reads from the YH dataset. For deBGA with $cl = 0.35$ setting, the corresponding numbers are 7037, 7148 and 14 592, respectively. This indicates the different design of the aligners, i.e. BWA-MEM favors local alignments, whereas deBGA aligns reads more globally.

3.2.2 Variant calling with the alignment of deBGA

Variant calling is one of the major applications of read alignment. We investigated the usability of deBGA on this application by real datasets. deBGA was implemented on a ~57× coverage NA12878 dataset (SRA accession numbers: ERR091571-74). The alignment result was used as input to call variants by GATK HaplotypeCaller (DePristo *et al.*, 2011). The called SNPs and indels were compared with the ground truth set of NA12878 provided by Genome in a Bottle (GIAB) Consortium (Zook *et al.*, 2014) and dbSNP ver. 146 (Sherry *et al.*, 2001). BWA-MEM was also implemented on the same dataset for comparison.

The numbers of the SNPs and indels called and matched by the GIAB ground truth sets and dbSNP are in Table 1. It is observed that, with the alignments of deBGA and BWA-MEM, the numbers of called SNPs passed the VQSR filter of GATK are within normal range, and they are comparable to each other. With deBGA, the pipeline recovered slightly more ground truth SNPs, indicating that the sensitivity of the deBGA callset could be better. The difference on the indels is more obvious, i.e. with deBGA, about 5% more indels recorded in the ground truth set were recovered. This could be due to that deBGA prefers end-to-end alignment, i.e. this strategy may produce more consecutive alignments for the reads having relatively large indels. In this situation, the indels would be involved in the alignment to be informative to variant callers. However, it is also very worthnoting that this difference only exists for a very small proportion of the reads and the variant calling results are comparable. Meanwhile, it is also observed that the two aligners have comparable numbers and proportions of VQSR-pass variants matched by dbSNP as well. This also indicates that quality of the callsets of two aligners could be similar to each other.

We also input the alignments of deBGA and BWA-MEM to state-of-the-art callers to call short tandem repeats (STRs), inversions and duplications, which are more difficult tasks in genome re-sequencing studies.

The Allelotype module of lobSTR (Gymrek *et al.*, 2012) was employed to call the STRs. The result (Supplementary Table S15) shows that comparable numbers of STRs were obtained based on the alignments of deBGA and BWA-MEM. We further compared the two callsets with the STR callset of 1000 Genomes Project Phase1 data produced by the lobSTR pipeline itself (available at: http://melissagymrek.com/lobstr-code/download.html). It is observed that deBGA and BWA-MEM also have similar numbers of called STRs which can be matched to the 1000 Genomes Phase1 callset. Considering the numbers of called and matched STRs, the two callsets are also likely to have similar quality, like that of SNPs and indels.

Delly (Rausch *et al.*, 2012) was employed to call two types of non-co-linear SVs, i.e. duplications and inversions. The results (Supplementary Table S16) showed that similar numbers of inversions were called with deBGA and BWA-MEM; and the number of duplications called with deBGA is higher than that of BWA-MEM, but they are still on the same magnitude. We manually checked a set of duplications in the callset of deBGA but not in that of BWA-MEM, and found that this issue is mainly caused by that deBGA aligned some of the reads as disconcordant pairs; however, BWA-MEM aligned them as normal pairs but with relatively larger edit distances (an example is in Supplementary Fig. S10). This may be due to the different design of the two aligners. Further, we compared the calls of the two aligners with the SV callsets of NA12878 provided by 1000 Genomes Project Release3 (Sudmant *et al.*, 2015), and found that, with deBGA and BWA-MEM, similar numbers of variants in the 1000 Genomes Project callset are recovered. This result indicates that the alignments of deBGA and BWA-MEM may make the downstream SV caller have similar sensitivities. Moreover, similar numbers of called SVs are matched as well (the reciprocal 50% criterion is used for determining the numbers of recovered and matched SVs). Considering these similar numbers, the quality of the callsets of deBGA and BWA-MEM is still comparable to each other. Moreover, there are still a large proportion of the variants called by deBGA and BWA-MEM cannot be matched by the 1000 Genomes Project callset, indicating that, for both of the two aligners, there are some SV-spanning reads still not well-handled. More advanced approach could be still needed to improve the quality of the alignment of SV-spanning reads.

3.3 Additional results

Other features of deBGA, including the time of index construction, the memory footprint and the performance of multiple threads, were also investigated. Refer to Supplementary Notes for these additional results.

4 Discussion

Results on various datasets demonstrated that deBGA can align reads to one or more genomes quickly and sensitively, making it particularly well-suited to handle the rapidly growing volumes of genomics and metagenomics data. This advantage originates from its de Bruijn graph-based organization of references and specifically designed seed-and-extension process, which enables to well handle the repeats. deBGA is suited to be integrated into many re-sequencing pipelines. Moreover, this method could be further extended to solve other important problems, which are also future works for us.

One of the potential works is to extend the approach to better align reads spanning the breakpoints of large or non-co-linear events, such as the DNA-seq reads spanning SVs, or RNA-seq reads which usually have splicing events. The alignment of such reads is difficult, and critical to many downstream analyses. Graph-based approach may have its own advantage to this task, as the non-co-linear events could be better modeled under the de Bruijn graph framework, comparing to the linear representation of the reference. However, the alignment method still needs to be well-designed to fully take the advantages of the graph representation.

Another potential extension is to use the deBGA method to align reads against other data structures which can also be split into *k*-mers,

such as MSA-based graphs. This will make deBGA an important step toward rapidly aligning reads to variation-aware references representing a population of genomes, such as the 1000 Genomes Project or the UK100K (http://www.genomicsengland.co.uk/) datasets.

Acknowledgements

We are very grateful to Orion Buske and Yue Jiang for their helpful discussion on the analysis of data and careful correction on the English writing. We also thank the reviewers of the paper for their very helpful comments and suggestions.

Funding

This work was partially supported by the National Nature Science Foundation of China [61301204 and 31301089], the National High-Tech Research and Development Program (863) of China [2015AA020101, 2015AA020108 and 2014AA021505] and the National Science and Technology Major Project [2013ZX03005012].

Conflict of Interest: none declared.

References

1000 Genomes Project Consortium (2012) An integrated map of genetic variation from 1,092 human genomes. *Nature*, **491**, 56–65.

Breitwieser,F.P. *et al*. (2015) Re-analysis of metagenomic sequences from acute flaccid myelitis patients reveals alternatives to enterovirus D68 infection. *F1000Res*., **4**, 180.

DePristo,M.A. *et al*. (2011) A framework for variation discovery and genotyping using next-generation DNA sequencing data. *Nat. Genet*., **43**, 491–498.

Didelot,X. *et al*. (2012) Transforming clinical microbiology with bacterial genome sequencing. *Nat. Rev. Genet*., **13**, 601–612.

Dilthey,A. *et al*. (2015) Improved genome inference in the MHC using a population reference graph. *Nat. Genet*., **47**, 682–688.

Dobin,A. *et al*. (2012) STAR: ultrafast universal RNA-seq aligner. *Bioinformatics*, **29**, 15–21.

Döring,A. *et al*. (2008) SeqAn: an efficient, generic C++ library for sequence analysis. *BMC Bioinformatics*, **9**, 11.

Fricke,W.F. and Rasko,D.A. (2014) Bacterial genome sequencing in the clinic: bioinformatic challenges and solutions. *Nat. Rev. Genet*., **15**, 49–55.

Gan,X. *et al*. (2011) Multiple reference genomes and transcriptomes for Arabidopsis thaliana. *Nature*, **477**, 419–423.

Gnerre,S. *et al*. (2011) High-quality draft assemblies of mammalian genomes from massively parallel sequence data. *Proc. Natl. Acad. Sci. USA*, **108**, 1513–1518.

Gymrek,M. *et al*. (2012) lobSTR: a short tandem repeat profiler for personal genomes. *Genome Res*., **22**, 1154–1162.

Huang,L. *et al*. (2013) Short read alignment with populations of genomes. *Bioinformatics*, **29**, i361–i370.

Kehr,B. *et al*. (2014) Genome alignment with graph data structures: a comparison. *BMC Bioinformatics*, **15**, 99.

Landau,G. and Vishkin,U. (1986) Introducing efficient parallelism into approximate string matching and a new serial algorithm. In: *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pp. 220–230. Assn for Computing Machinery, Berkeley, CA, USA.

Langmead,B. and Salzberg,S.L. (2012) Fast gapped-read alignment with Bowtie 2. *Nat. Methods*, **9**, 357–359.

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

Li,H. and Durbin,R. (2010) Fast and accurate long-read alignment with Burrows-Wheeler transform. *Bioinformatics*, **26**, 589–595.

Li,H. and Homer,N. (2010) A survey of sequence alignment algorithms for next-generation sequencing. *Brief Bioinform*., **11**, 473–483.

Li,H. (2013) Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. arXiv:1303.3997.

Marco-Sola,S. *et al*. (2012) The GEM mapper: fast, accurate and versatile alignment by filtration. *Nat. Methods*, **9**, 1185–1188.

Medini,D. *et al*. (2005) The microbial pan-genome. *Curr. Opin. Genet. Dev*., **15**, 589–594.

Mu,J.C. *et al*. (2012) Fast and accurate read alignment for resequencing. *Bioinformatics*, **28**, 2366–2373.

Paten,B. *et al*. (2014) Mapping to a Reference Genome Structure. arXiv:1404.5010.

Pruitt,K.D. *et al*. (2014) RefSeq: an update on mammalian reference sequences. *Nucleic Acids Res*., **42**, D756–D763.

Raphael,B. *et al*. (2004) A novel method for multiple alignment of sequences with repeated and shuffled elements. *Genome Res*., **14**, 2336–2346.

Rausch,T. *et al*. (2012) DELLY: structural variant discovery by integrated paired-end and split-read analysis. *Bioinformatics*, **28**, i333–i339.

Schneeberger,K. *et al*. (2009) Simultaneous alignment of short reads against multiple genomes. *Genome Biol*., **10**, R98.

Sherry,S.T. *et al*. (2001) dbSNP: the NCBI database of genetic variation. *Nucleic Acids Res*., **29**, 308–311.

Siren,J. *et al*. (2014) Indexing graphs for path queries with applications in genome research. *IEEE/ACM Trans. Comput. Biol. Bioinform*., **11**, 375–388.

Sudmant,P.H. *et al*. (2015) An integrated map of structural variation in 2,504 human genomes. *Nature*, **526**, 75–81.

Treangen,T.J. and Salzberg,S.L. (2012) Repetitive DNA and next-generation sequencing: computational challenges and solutions. *Nat. Rev. Genet*., **13**, 36–46.

Valenzuela,D. *et al*. (2015) On enhancing variation detection through pan-genome indexing. bioRxiv, doi: 10.1101/021444.

Wang, J. *et al*. (2008) The diploid genome sequence of an Asian individual. *Nature*, **456**, 60–65.

Zook,M.Z. *et al*. (2014) Integrating human sequence data sets provides a resource of benchmark SNP and indel genotype calls. *Nat. Biotechnol*., **32**, 246–251.