BWTaligner: a genome short-read aligner

Lam Nguyen[1], Xuan Thi Trinh[2], Hien Trinh[3], Dang Hung Tran[4], Cuong Nguyen[1*]

[1]*Vinmec Research Institute of Stem Cell and Gene Technology*
[2]*Faculty of Information Technology, Hanoi Open University*
[3]*Laboratory of Genetic Engineering, Institute of Biotechnology, Vietnam Academy of Science and Technology*
[4]*Hanoi National University of Education*

Received 6 April 2018; accepted 15 May 2018

Abstract:

The development of next-generation sequencing technologies has helped sequence large genomes easily, producing a huge number of short-reads - small fragments of DNA. Despite the existence of many developed alignment tools, mapping short-read datasets to the reference genome, a crucial step of genome analysis, still remains a challenge. In this study, we develop a short-read alignment program, BWTaligner, based on the Burrows-Wheeler transform compression - exact and inexact matching. We tested it on the paired-end read data simulated from chromosome 9 of the rice genome to compare the alignment and single-nucleotide polymorphism (SNP) calling between our aligner and BWA - the preferred alignment program. The results showed that the BWA delivers higher recall and F-score, while BWTaligner has better precision in high coverage depth.

Keywords: ***Burrows-Wheeler transform, high-throughput sequencing, paired-end short reads, sequence alignment.***

Classification number: ***3.5***

Introduction

The development of massive parallel sequencing technologies has stimulated the production of a vast number of short-reads, which are small fragments of DNA genomes. As the mapping of short-read datasets to large genomes presents a huge challenge to the existing sequencing programs, more and more algorithms are being improved in order to reduce the execution time and increase the mapping accuracy. At the outset, hash table-based methods either hash the short-read sequences or the reference genome and many alignment tools have been developed to resolve this. The aligners based on hashing short reads are typically MAQ [1], ZOOM [2], and SHRiMP [3]. MAQ is one of the old programs that supports ungapped sequence alignments and shown quality scores, while ZOOM limits a number of mismatches. SHRiMP indexes both the short-reads and the genome. These aligners have a flexible memory footprint, which have been capable of overhead when a small number of reads are mapped. The tools hashing the genome, such as SOAP 1 [4], PASS [5], MOM [6], mrFAST/mrsFAST [7], and BFAST [8] can be parallelized using numerous threads; however, they need a large memory to build an index for the reference genome. Interestingly, mrFAST/mrsFAST employs a seed-and-extend strategy that initially identifies candidate positions for a short-read and then uses different alignment algorithms, such as the Smith-Waterman algorithm [9], for mapping. In addition to initial hash table-based methods, the other alignment algorithm is a slider that merges and sorts the reference subsequences

*Corresponding author: Email: v.cuongn@vinmec.com

and short-reads.

As the alignment algorithms using a hash table often require a large amount of memory, new alignment programs based on suffix/prefix tries were generated to reduce the memory requirements. The suffix/prefix tries perform backward searched and the Burrows-Wheeler Transform (BWT) [10] for exact matching, which has led to the development of several aligners, including Bowtie [11], SOAP 2 [12], and BWA [13]. Furthermore, they also provide support for paired-end alignment. Bowtie, including Bowtie 1 and 2 [14], is one of the first programs to use FM-index [15, 16], which is built on the BWT and mimics backward search. For reads shorter than about 50 bp, Bowtie 1 is sometimes more sensitive, while Bowtie 2 supports gapped alignment and works better for longer short-reads. SOAP 2 combines the hashing and FM-index to speed up but uses more memory than BWA and Bowtie. The efficiency of the BWA aligner for inexact matching is widely known, and it is still used by researchers. All of these aligners are fast and have been optimized for multi-core Central Processing Units (CPUs). However, increasing the speed of alignment process provides time saving, especially with regard to processing large-scale data; hence, a multiple-core Graphics Processing Units (GPUs)-based method is a powerful choice. There are several alignment tools based on GPU, including SOAP3 [17] and BarraCUDA [18].

In this research, the introduction of BWTaligner based on the BWT algorithm, exact and inexact matching, has been made. Moreover, we have evaluated the performance of BWTaligner on simulated data by comparing it with BWA in single-nucleotide polymorphism (SNP) calling - the finding corresponding to the variations occurring in the genome.

Materials and methods

Burrow-Wheeler transform

The BWT construction reduces the execution speed and memory in the running process. Let G be a reference genome sequence that is constructed by four nucleotides (A, C, G, T). The symbol $ is lexicographically smaller than all the characters in G and only appears at the end to form a new sequence, G$. The matrix M, which is built from the rotations of G$, is sorted by lexicographical order, and each column is a permutation of G$. The transformed B can be attained by taking the last column of matrix M. A suffix array (SA) is defined as an array of integers with the starting position of the i-th smallest suffix of G. This algorithm is illustrated in Fig. 1.

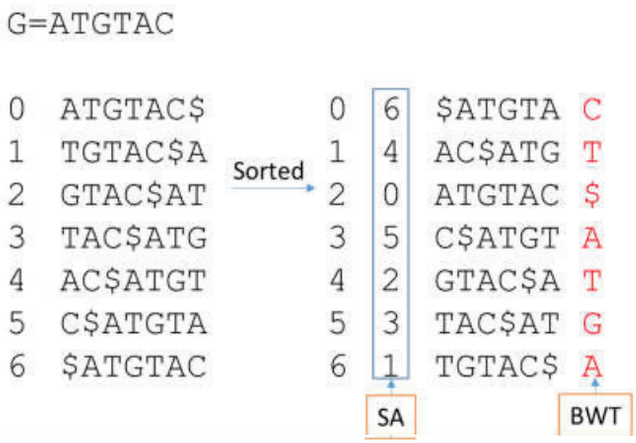

Fig. 1. The BWT construction and suffix array of reference genome G = ATGTAC. BWT matrix includes seven rows that are in lexicographical order. Forward BWT is defined as CT$ATGA and SA is (6, 4, 0, 5, 2, 3, 1).

Exact matching

Let Q be a query sequence that is a substring of reference genome G. A backward search based on FM index [15] was used to find each occurrence of Q (Fig. 2), which is actually the search for an SA interval. Oc(α,i) is the number of occurrences of α in B[0,i]. C(α) is the number of symbols in Q[0,n-2] that are lexicographically smaller than α ϵ G. Ferragina and Manzini showed the SA interval for searching for all occurrences of Q in G using the forward BWT as follows:

$$Rs = C(Q[i]) + O\ (Q[i], Rs(i+1) - 1) + 1, \quad i \in [0, |Q|]$$
$$Re = C(Q[i]) + Oc\big(Q[i], Re(i+1)\big), \quad i \in [0, |Q|]$$

where: Rs and Re indicate the start and end of the SA interval, respectively. Q is a substring of G if and only if Rs(i) ≤ Re(i). However, as the total array of Oc is sorted, more memory and execution time are required. For reducing

the memory footprint of the Oc array, only a part of the Oc is stored and calculated using the length of Oc (L).

```
Backward_Search(Q[1,q])
{
     i = q, c = Q[q], First = C[c]+1, Last = C[c+1];
     while ( ( First ≦ Last) and i ≧ 2 )
     {
          c = Q[i-1];
          First = C[c] + Oc(c, First-1)+1;
          Last  = C[c] + Oc(c, Last); i = i-1;
     }
     if (Last < First) then
          return "no occurrence" ;
     else
          return ( First, Last );
}
```

Fig. 2. The pseudocode of backward search.

Inexact matching

Sequencing errors and the differences between the sequence and reference genome are inexact matches. The inexact matches can be attained by comparing input sequences with the reference genome in order to identify variations, such as substitutions, insertions, and deletions. However, the exact matching only provides ungapped alignment; therefore, insertions and deletions were not allowed. Hence, inexact match searching could be converted to exact matches based on all the permutations of short reads. A total of permutations can be performed by the 4-ary tree, where each permutation represents different routes (Fig. 3).

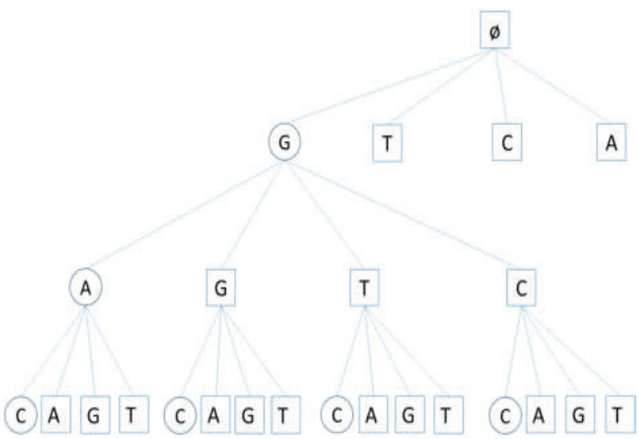

Fig. 3. A 4-ary tree example for searching the inexact matches of sequence "GAC" using BWT. The circles are defined as the original bases and rectangles as the mutated bases.

Each node denotes a base in the query with the same position. Currently, there are two approaches to searching for all inexact matches - depth-first search (DFS) and breadth-first search (BFS). The BFS approach requires a large memory capacity for storing all the results, so this approach is impractical for GPU computing. Our aligner implemented the DFS approach, where the memory expenditure is small and equivalent to the size of the tree. Nevertheless, recursive functions are still supported for the Fermi architecture [19]. Fig. 4 illustrates the pseudocode of inexact matching. The number of inexact matching of sequences can be estimated through calculating the number of bases that do not exactly match the genome. z(*) is defined as the full length of the query sequence (Q), where z(i) is defined as the number of inexact matches in the query Q[i + 1, |Q|-1] ($0 \leq i \leq |Q| -1$). For seed alignment, zw(*) is calculated, where zw(i) represents the number of

```
# G: genome, Q: query sequence
# bwt and rbwt are forward and reverse BWT of G
1. Searching for inexact matches
Inexact_matches_search_full_length(Q,G,f,l,z)
{
   z=0;f=0;l=|G|
   for i = |Q|-1 to 0;
   {
      f = bwt.C(Q[i], f-1)+1;l = bwt.C(Q[i]) + bwt.Oc(Q[i],l); z[i]=z
   }
      if f > l then
         f = 0; l = |G|; ++z;
         return "inexact matches";
}
Inexact_matches_search_seed_length(W, G, f, l, z)
# W: seed length (a part of Q)
{
   zw=0, f=0; l = |G|
   for i = W-1 to 0
   {
      f = bwt.C(Q[i], f-1)+1; l = bwt.C(Q[i]) + bwt.Oc(Q[i],l); zw[i]=z
      if f > l then
         f = 0; l = |G|; ++z;
         return "inexact matches";
   }
}
2. The initialization of the stack from the first base of Q()
Initialization_stack_first_base()
{
   while (stack is not empty)
   {
      access to current node (top node)
      if there is not mutation on current node then
         continue;
      fi
      there is mutation in the current node
      calculate SA interval (f,l)
      if f <= l then
         if kepping to the end of Q then
            return "a hit is found";
         else
            push next base in Q into the stack;
         fi
      fi
   }
}
```

Fig. 4. The pseudocode for inexact matching.

substitutions mismatching correctly to the substring Q[i+1, W-1] ($0 \leq i \leq W-1$).

Results and discussion

We evaluated the performance of the BWTaligner by comparing it to BWA version 0.6.2, the alignment tool using simulated reads that is most widely used. The paired-end datasets were simulated from chromosome 9 in the reference rice genome, Nipponbare version 7.0 (Genbank accession number PRJDB1747, 23,012,720 bp) using *wgsim*, a short-read simulator (version 0.3.1) with a different depth of coverage, including 5X, 10X, and 30X 100 bp-paired-end reads; 0.085% mutation rates (19,560 SNPs); and 0.02% base error rates. First, we evaluated the alignment quality of each tool and subsequently called SNPs from aligned short reads using *mpileup* in SAMtools and VarScan. Finally, the identified SNPs were compared with the simulated SNPs. All the tests were carried out on a workstation with a X5650 @ 2.67 GHz 24-core processor and 198 GB RAM running the Ubuntu 14.10.

Table 1 showed that over 99.0% of simulated paired-end reads aligned with the reference genome. With regard to the number of aligned reads, BWA was slightly higher at three depths of coverage. The SNP calling performance of the aligners was evaluated using the precision, recall, and F-score (Table 2). Precision is defined as TP/(TP+FP), recall as TP/(TP+FN), and F-score as 2*precision*recall/(precision+recall), where TP is a true positive, that is, the number of correct SNPs. FP is a false positive, performing the mismatch, and FN is a false negative, representing that the simulated SNPs were not determined along with the missed SNP. From Table 3, it can be seen that at the lower coverage (5X and 10X), BWA has better precision, while at 30X depth, the precision of BWTaligner is higher (99.16% as compared to 99.03% of BWA). While the precision is a positive predictive value and based on the number of positive SNPs, the recall, i.e., the sensitivity, is considered as the number of negative SNPs and F-score value, which is the harmonic mean of the precision and recall. The results of our study showed that BWA always leads to a higher recall and F-score than BWTaligner at all coverages. Furthermore, F-scores tend to increase as the depth of coverage gets higher. This denotes that the depth of coverage plays an important role in the accuracy of alignment and SNP calling.

Table 1. Alignment of simulated reads.

Depth of coverage	Stimulated paired-end reads	Aligned reads using BWA (%)	Aligned reads using BWTaligner (%)
5X	575,318	99.57	99.38
10X	1,150,636	99.58	99.41
30X	3,451,908	99.58	99.41

Table 2. Performance of SNP calling under different coverage.

		BWA		BWTaligner	
Call SNP at 5X	TP	1,182	6.01%	891	4.55%
	FP	3	0.02%	9	0.05%
	FN	18,468	93.97%	18,669	95.40%
Call SNP at 10X	TP	9,439	47.98%	8,223	41.92%
	FP	21	0.11%	58	0.30%
	FN	10,211	51.91%	11,337	57.79%
Call SNP at 30X	TP	19,155	96.56%	18,951	96.10%
	FP	187	0.94%	161	0.82%
	FN	495	2.50%	609	3.09%

Table 3. Precision, recall and F-measure between BWA and BWTaligner.

	BWA			BWTaligner		
	5X	*10X*	*30X*	*5X*	*10X*	*30X*
Precision	0.9974	0.9978	0.9903	0.9900	0.9930	0.9916
Recall	0.0601	0.4804	0.9748	0.0456	0.4204	0.9689
F-score	0.1134	0.6485	0.9825	0.0871	0.5907	0.9801

Conclusions

We preliminarily present the BWTaligner based on the basic algorithms, including the Burrows-Wheeler Transform, backward search for exact and inexact matching. This tool will be further developed and empirically studied so as to address the short-read alignment challenge with regard to time and accuracy.

REFERENCES

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

[2] H. Lin, Z. Zhang, M.Q. Zhang, B. Ma, M. Li (2008), "ZOOM! Zillions of oligos mapped", *Bioinformatics*, **24**(**21**), pp.2431-2437.

[3] S.M. Rumble, P. Lacroute, A.V. Dalca, M. Fiume, A. Sidow, M. Brudno (2009), "SHRiMP: accurate mapping of short color-space reads", *PLoS Computational Biology*, **5**(**5**), pp.e1000386.

[4] R. Li, Y. Li, K. Kristiansen, J. Wang (2008), "SOAP: short oligonucleotide alignment program", *Bioinformatics*, **24**(**5**), pp.713-714.

[5] D. Campagna, A. Albiero, A. Bilardi, E. Caniato, C. Forcato, S. Manavski, G. Valle (2009), "PASS: a program to align short sequences", *Bioinformatics*, **25**(**7**), pp.967-968.

[6] H.L. Eaves, Y. Gao (2009), "MOM: maximum oligonucleotide mapping", *Bioinformatics*, **25**(**7**), pp.969-970.

[7] F. Hach, F. Hormozdiari, C. Alkan, F. Hormozdiari, I. Birol, E.E. Eichler, S.C. Sahinalp (2010), "mrsFAST: a cache-oblivious algorithm for short-read mapping", *Nature Methods*, **7**(**8**), pp.576-577.

[8] N. Homer, B. Merriman, S.F. Nelson (2009), "BFAST: an alignment tool for large scale genome resequencing", *PLOS ONE*, **4**(**11**), pp.e7767.

[9] R. Mott (2005), "Smith-Waterman Algorithm", *Encyclopedia of Life Sciences*, Chichester: John Wiley & Sons, Ltd.

[10] M. Burows, D.J. Wheeler (1994), *A block sorting lossless data compression algorithm*, CA, Digital Equipment Corporation.

[11] B. Langmead, C. Trapnell, M. Pop, S.L. Salzberg (2009), "Ultrafast and memory-efficient alignment of short DNA sequences to the human genome", *Genome Biology*, **10**(**3**), pp.R25.

[12] R. Li, C. Yu, Y. Li, T.W. Lam, S.M. Yiu, K. Kristiansen, J. Wang (2009), "SOAP2: an improved ultrafast tool for short read alignment", *Bioinformatics*, **25**(**15**), pp.1966-1967.

[13] H. Li, R. Durbin (2009), "Fast and accurate short read alignment with Burrows-Wheeler transform", *Bioinformatics*, **25**(**14**), pp.1754-1760.

[14] B. Langmead, S.L. Salzberg (2012), "Fast gapped-read alignment with Bowtie 2", *Nature Methods*, **9**(**4**), pp.357-359.

[15] P. Ferragina, G. Manzini (2005), "Indexing compressed text", *Journal of the ACM*, **52**(**4**), pp.552-581.

[16] P. Ferragina, G. Manzini (2000), "Opportunistic data structures with applications", *Proceedings of the 41st annual symposium on foundations of computer science*, pp.390-398, IEEE.

[17] C.M. Liu, T. Wong, E. Wu, R. Luo, S.M. Yiu, Y. Li, T.W. Lam (2012), "SOAP3: ultra-fast GPU-based parallel alignment tool for short reads", *Bioinformatics*, **28**(**6**), pp.878-879.

[18] P. Klus, S. Lam, D. Lyberg, M. Cheung, G. Pullan, I. McFarlane, B.Y. Lam (2012), "BarraCUDA - a fast short read sequence aligner using graphics processing units", *BMC Research Notes*, **5**(**1**), p.27.

[19] E. Lindholm, J. Nickolls, S. Oberman, J. Montrym (2008), "NVIDIA Tesla: a unified graphics and computing architecture", *IEEE Micro*, **28**(**2**), pp.39-55.